

Planning non-holonomic stable trajectories on uneven terrain through non-linear time scaling

Arun Kumar Singh¹ · K. Madhava Krishna¹ · Srikanth Saripalli²

Received: 3 January 2014 / Accepted: 23 September 2015 / Published online: 28 November 2015
© Springer Science+Business Media New York 2015

Abstract In this paper, we present a framework for generating smooth and stable trajectories for wheeled mobile robots moving on uneven terrains. Instead of relying on static stability measures, the paper incorporates velocity and acceleration based constraints like no-slip and permanent wheel ground contact conditions in the planning framework. The paper solves this complicated problem in a computationally efficient manner with full 3D dynamics of the robot. The two major aspects of the proposed work are: Firstly, closed form functional relationships are derived which describes the 6 *dof* evolution of the robot's state on an arbitrary 2.5D uneven terrain. This enables us to have a fast evaluation of robot's dynamics along any candidate trajectory. Secondly, a novel concept of *non-linear time scaling* is introduced through which simple algebraic relations defining the bounds on velocities and accelerations are obtained. Moreover, *non-linear time scaling* also provides a new approach for manipulating velocities and accelerations along given geometric paths. It is thus exploited to obtain stable velocity and acceleration profiles. Extensive simulation results are presented to demonstrate the efficacy of the proposed methodology.

Keywords Uneven terrain navigation · Non-holonomic motion planning · Dynamic stability · Time scaling

✉ Arun Kumar Singh
aks1812@gmail.com; arunkumar.singh@research.iiit.ac.in

K. Madhava Krishna
mkrishna@iiit.ac.in

Srikanth Saripalli
srikanth.saripalli@asu.edu

¹ The Robotics Research Centre IIIT, Hyderabad, India

² ASTRIL, Arizona State University, Phoenix, AZ, USA

1 Introduction

Trajectory generation for mobile robots on uneven terrain is a challenging problem because the notion of the robot stability governed by the robot-terrain interaction has to be included in the planning framework. Throughout this paper, a mobile robot will be understood to be stable if its velocity and acceleration profile satisfies the no-slip and permanent wheel ground contact conditions. As shown in our previous work, Singh et al. (2011), Singh and Krishna (2013), that checking for satisfaction of these constraints provide a more correct picture of robot stability than some other measures like Tip-Over which primarily only depends on the variation of robot's pitch and roll angle. However, generating a trajectory between a given start and goal state which explicitly satisfies the no-slip and contact constraints is associated with some fundamental complexities.

Firstly, the variation of robot's position and posture as it evolves on the uneven terrain needs to be ascertained. For a passive suspension robot, only the yaw plane parameters can be controlled. The evolution of the rest of the states like pitch and roll angle depends on the yaw plane parameters and the underlying terrain. Thus, to model the robot dynamics in 3D it is imperative to obtain mathematically these functional relationships. Secondly, as shown in later sections the no-slip and the permanent wheel ground contact constraints can be written in terms of traction and normal wheel ground contact forces satisfying certain inequality constraints. But the inverse dynamic relations which relates traction and normal forces to the robot's velocity and acceleration profile, it's posture and the underlying terrain geometry are not known in closed form functional form.

1.1 Related work on uneven terrain trajectory generation

The problem of uneven terrain trajectory generation has been addressed by many researchers at various level in the past couple of decades. One of the earlier works in this field was proposed by Shiller and Gwo (1991) wherein no-slip and permanent wheel ground contact were used as constraints while generating smooth trajectories for car-like robots. Along the same lines Cherif (1999) proposed a two level trajectory generation framework which accounted for robot geometry and 3D posture. In it, at the first level smooth paths are generated considering flat terrain and thereafter feasible motion profiles are computed for the paths considering the effects of uneven terrain.

A popular methodology for uneven terrain motion planning has been to generate trajectories passing through such regions of the terrain which satisfy the stable threshold limit of roll, pitch angle and robot-ground clearance. One such work was proposed by Kubota et al. (2001) wherein stable paths were computed, though without imposing any kind of kinematic constraints on the paths. Moreover, calculation of motion profiles corresponding to the stable paths was not presented. Similar limitations also exist in a recent work proposed by Miro et al. (2010). An improved methodology was proposed by Bonnafous et al. (2001), wherein the planner directly worked at the control input level (velocity or acceleration) and produced trajectories satisfying the limit on pitch and roll angle.

A comprehensive work on uneven terrain trajectory generation was presented by Howard and Kelly (2007). Using a parametric optimal control framework, curvature continuous trajectories were generated connecting arbitrary initial and final states. A major novelty of the work lies in the fact that it includes the effect of uneven terrain while computing motion profiles for the trajectories. Moreover, effects of terrain dependent phenomena like wheel-slip were also included in the trajectory generation framework.

A trajectory generation using point mass robot model was proposed by Iagnemma et al. (2008) wherein a navigation framework using potential field was also incorporated. A trajectory generation framework considering planar robot model on planar sloped terrains was presented by Waheed and Fotouhi (2009). A constant time scaling concept similar to Hollerbach (1984) was used to generate stable motion profiles for the robot.

1.2 Contributions

The proposed work presents an in-depth analysis of the factors affecting the dynamic stability of a wheeled mobile robot on a 2.5D uneven terrain. As stated earlier, a major challenge in uneven terrain motion planning is to ascertain the 6 *dof*

evolution of the robot's state. We present a novel approach where we derive robot's 6 *dof* evolution on terrain by coupling the geometric holonomic constraints of the robot and the surface equation. As described in our earlier work Singh et al. (2011), the resulting set of coupled non-linear equations can be solved numerically. In the current work, we complement the numerical solution by providing approximate closed form solutions to the set of non-linear equations describing the robot's 6 *dof* state. As explained later, a trade-off can be reached between opting for approximate closed form and exact numerical solution. This greatly reduces the computational complexity and enables us to have fast evaluation of robot's dynamics along any candidate trajectory. The robot's state given by the approximate relations are shown to agree well with that obtained through exact solution of non-linear equations for variety of terrains considered in our simulations.

As evident, the proposed work shares conceptual similarity with Shiller and Gwo (1991), Cherif (1999), Iagnemma et al. (2008) and Waheed and Fotouhi (2009) in the fact it also considers permanent wheel-ground contact and no-slip as the stability constraints. However, the proposed work goes beyond these cited works by incorporating a complete 3D model of robot dynamics. This is in contrast to the point mass model of Shiller and Gwo (1991), Iagnemma et al. (2008), and planar motion equations used in Cherif (1999) and Waheed and Fotouhi (2009). Although Mann et al. 2008 proposed a dynamic stability metric considering quasi-3D robot model, it was never used within a planning framework. A motion planning framework based on the stability analysis proposed by Mann et al. would require ascertaining the stability of each candidate trajectory separately in the roll, pitch and yaw plane. This would require computing the robot dynamics and stable velocity and acceleration profiles thrice. Since the proposed framework works with the full 3D dynamics, the stability computations only needs to be performed once for any given candidate trajectory. Moreover, as shown later the stability computations reduces to computing the intersection of single variable quadratic inequalities.

The proposed work also introduces a new concept called *non-linear time scaling* for manipulating velocity and acceleration along given geometric paths. Conventionally, fitting velocity and acceleration profiles to a geometric path requires parametrizing the path in terms of arc length and then computing the arc length velocities and accelerations (Shiller and Gwo 1991; Mann et al. 2008). For higher order curves like spline, the cartesian coordinate variables (x , y , z) and the arc length are related through a non-analytical integral (Wang et al. 2002). Since, the terrain equation and the robot's 6 *dof* evolution model are expressed as a function of Cartesian coordinate variables, evaluating the robot's state and dynamics at various arc lengths would involve repeated computation of the numerical integral of arc length parametrization and

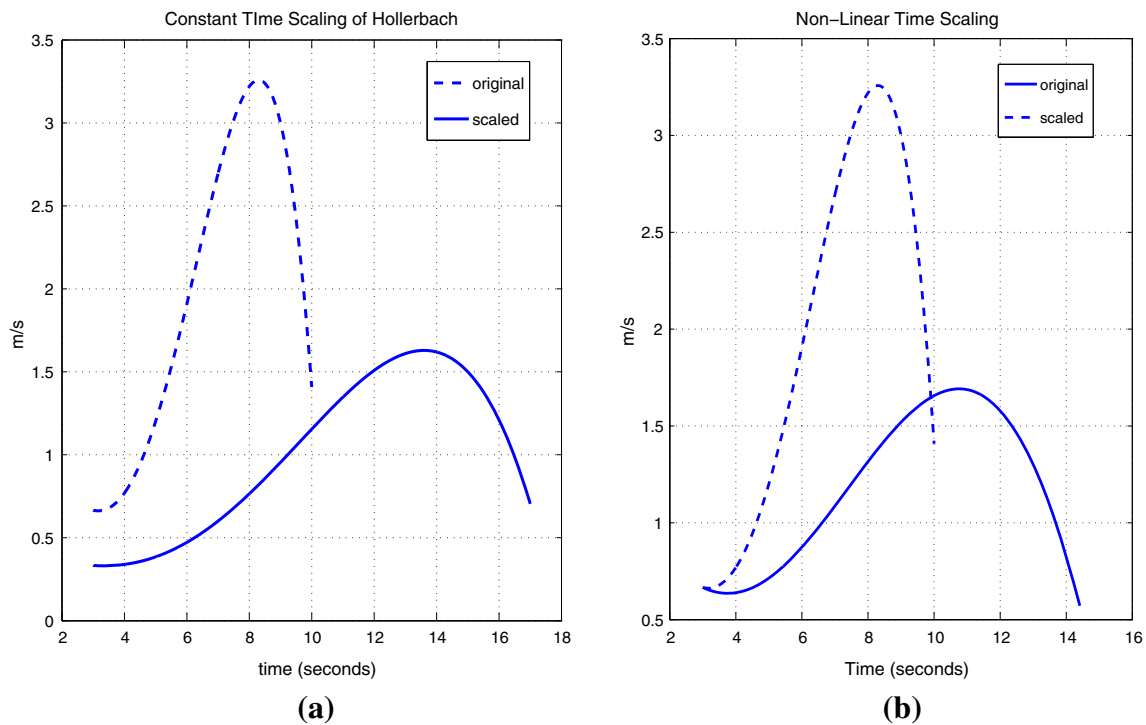


Fig. 1 Problem with constant scaling. It should be noted how the initial and scaled velocities does not start at the same point in **a**. This results in a discontinuity at the velocity level when constant scaling is employed. This is the main reason why constant scaling transformation cannot be

used in a reactive sense i.e it is not possible to employ constant scaling transformation while executing a trajectory. Non-linear time scaling solves this problem as shown in **b**

would add to the computational cost of the motion planning framework. These fundamental limitations form the major motivation behind introducing *non-linear time scaling* for uneven terrain motion planning. As shown later, *non-linear time scaling* provides simple algebraic relations for fitting velocity and acceleration profiles to a given geometric path and is exploited elegantly in the proposed framework to obtain stable trajectories.

1.3 Assumptions, formal problem statement and technical approach

The proposed framework works with the assumption that the terrain knowledge in the form of terrain equation is known. Some works related to fitting terrain equations to Digital Elevation Models can be found in [Bajaj et al. \(1995\)](#), [Amenti et al. \(1998\)](#) and [Prokos et al. \(2010\)](#).

With this assumptions in place, the problem addressed in the current work is of generating trajectories $\mathbf{X}(t) \in \mathbb{R}^6$ satisfying the following constraints

$$N_i = f_{1i}(\mathbf{X}(t), \dot{\mathbf{X}}(t), \ddot{\mathbf{X}}(t)) > 0 \tag{1}$$

$$|(T_i)| - \rho N_i = f_{2i}(\mathbf{X}(t), \dot{\mathbf{X}}(t), \ddot{\mathbf{X}}(t)) \leq 0 \tag{2}$$

Here N_i and T_i are the normal wheel-ground contact and wheel-ground traction forces respectively at the i^{th} wheel.

The technical approach followed here is to first find a parametric function $\mathbf{X}(u)$ as a function of some arbitrary variable u connecting the initial and goal states. Thereafter the transformation from the variable u to time variable, t is brought such that

$$\frac{du}{dt} = \dot{u} = g(\cdot) \tag{3}$$

The time dependent trajectory obtained below as a result of the scaling transformation should satisfy (1) and (2).

$$\begin{aligned} \dot{\mathbf{X}}(t) &= \mathbf{X}'(u) \frac{du}{dt}, \\ \ddot{\mathbf{X}}(t) &= \left(\frac{du}{dt}\right)^2 \mathbf{X}''(u) + \mathbf{X}'(u) \frac{d^2u}{dt^2} \end{aligned} \tag{4}$$

The terms \dot{u} or $g(\cdot)$ will be referred to as the scaling function. The simplest form of scaling function can be expressed as

$$g(\cdot) = \dot{u} = c \tag{5}$$

Here c is a scalar constant. This kind of scaling transformation was proposed by [Hollerbach \(1984\)](#). However, there is a critical problem with such transformations which severely

limits its use. To understand this further, suppose that some function $f(\cdot)$ shown in Fig. 1a describes the velocity of a robot and we employ the above scaling transformation to modify it. It can be seen that the scaled and unscaled velocity profile does not start at the same magnitude which means that the constant scaling will bring a discontinuity in the velocity space at the initial point. Another interpretation of this is that it is only possible to use constant scaling transformation at a planning level and not in a reactive manner while executing a trajectory. For example, midway while executing a trajectory, if it is required to change for any reason the velocity and acceleration profile while keeping the path unchanged, constant scaling of Hollerbach (1984) cannot be used. Obviously, the problem of velocity discontinuity does not arise when the initial unscaled velocity profile starts from zero which is exactly what is being used by Hollerbach (1984) and Waheed and Fotouhi (2009). However, on an uneven terrain a trajectory generation framework which requires the robot to first de-accelerate to zero velocity is not appropriate.

To overcome these problems the following scaling function is proposed. It can be seen from Fig. 1b that *non-linear time scaling* does not result in any velocity discontinuity as the scaled velocity start at the same point as the original.

$$\dot{u} = pe^{-qu} \quad (6)$$

Moreover, as shown later this kind of scaling transformation proves to be very versatile for computing velocity and acceleration profile. Although the term non-linear scaling has been used with piece-wise constant scaling in Waheed and Fotouhi (2009), it still suffers from the same drawback as the constant scaling in the sense that two different trajectories has to pass through the region of zero velocity at the connected point.

With the above described process a very efficient trajectory generation process has been proposed for non-holonomic robots. The framework builds upon our previous work, Singh et al. (2012) by providing simple algebraic inequalities governing the existence of stable motion profile for a given candidate path. Further, an optimization framework has been incorporated to maximize the velocity and acceleration within the stability limit along a candidate path.

1.4 System overview and layout of the paper

Figure 2 gives an overview of the framework proposed in the current work. First, a path in the yaw plane for the position x , y and the yaw angle α is generated connecting the start and the goal states. This yaw plane path is generated in the form of parametric functions in terms of an arbitrary variable u . Hence along with the path we also obtain x' , x'' and similar derivatives along the planned path. The yaw plane parameters in terms of x , y , α and their derivatives are then used to

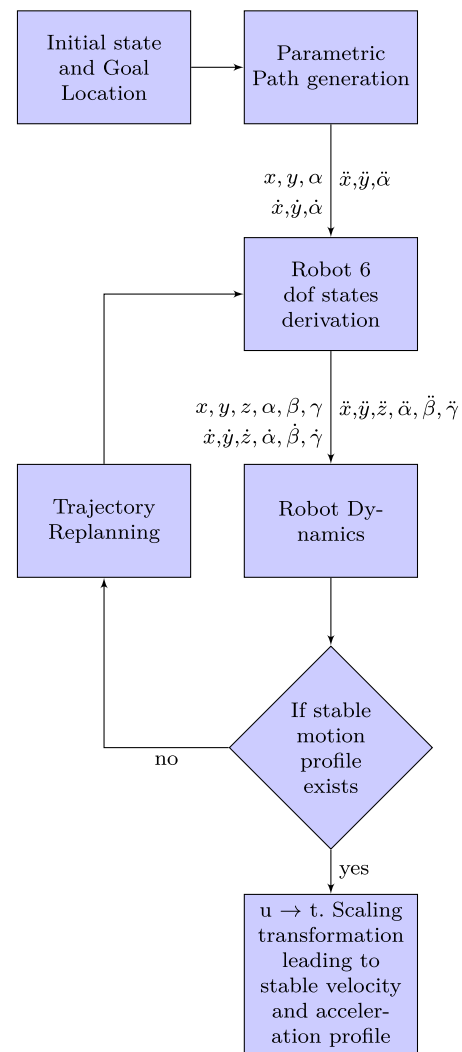


Fig. 2 System overview

estimate the full 6 *dof* states of the robot considering terrain conditions and robot geometry. Once the complete 6 *dof* state evolution along a path is obtained, it is used along with robot dynamics and dynamic stability constraints to compute the scaling transformation from the path variable u to the time variable t and consequently obtain the stable velocity and acceleration profile. If no scaling transformation leading to stable velocities and accelerations exist for a given path, trajectory replanning is evoked which re-plans and produces a new path from start/current state to the goal.

The parametric path generation module shown in the Fig. 2 is explained in Sect. 2. The derivation of robot 6 *dof* states and dynamics is provided in Sect. 3. Section 4 combines the derivations of Sects. 2 and 3 to derive a set of inequalities, the solution of which gives a scale factor. The scale factor information is then used in Sect. 5 which lays down the procedure for constructing the scaling function. The scaling function brings the transformation $u \rightarrow t$ and generates the

stable velocity and acceleration profile. Section 7 presents simulation results elucidating the proposed framework.

2 Parametric path generation

For a passive suspension robot, only the yaw plane components of the whole state space i.e position x , y , yaw α and their derivatives can be controlled (for the ease of notation, we do not show the dependency of state variables on time). All the other states of the robot are a function of yaw plane components and in later sections we derive these exact functional relationships. In this section the yaw plane paths are derived as parametric functions for a given a start and goal position, in similar lines with Nagy and Kelly (2001), Inanc et al. 2005, Guo et al. (2007). The paths are represented as a function of an arbitrary variable u . There are various choices for the form of parametric functions like polynomial (Guo et al. (2007), Fourier series like functions (Singh et al. 2012; Gallina and Gasperatto 2000) and splines (Shiller and Gwo 1991). Due to their numerical stability we use Bernstein polynomial based representation in the current work.

We start by parametrizing the path derivatives as

$$x'(u) = \sum_{i=0}^n B_i^n(u)a_i \tag{7}$$

$$\zeta(u) = \tan \alpha = \sum_{i=0}^n B_i^n(u)b_i \Rightarrow \alpha'(u) = \frac{\zeta'(u)}{1 + \zeta(u)^2} \tag{8}$$

where

$$B_i^n = \binom{n}{i} \left(1 - \frac{u - u_0}{u_f - u_0}\right)^{n-i} \left(\frac{u - u_0}{u_f - u_0}\right)^i, \forall u \in [u_0, u_f]$$

The non-holonomic constraint of the robot leads to the following relation

$$y' \cos \alpha - x' \sin \alpha = 0 \Rightarrow y' = x' \tan \alpha = x' \zeta \tag{9}$$

Integrating (7) and (9) we get the parametrised yaw plane path $(x(u), y(u), \alpha(u))$ The parameters a_i, b_i can be obtained by solving the following set of equations

$$\begin{cases} x(u_0) = x_0, y(u_0) = y_0, \zeta(u_0) = \zeta_0 \\ \frac{dx}{du}(u_0) = v_x, \frac{d\alpha}{du}(u_0) = \dot{\alpha} \\ x(u_f) = x_f, y(u_f) = y_f, \zeta(u_f) = \zeta_f \\ x(u_c) = x_c, y(u_c) = y_c \end{cases} \tag{10}$$

Here x_0, y_0, x_f, y_f are the initial and goal position. ζ_0, ζ_f corresponds to the initial and final heading constraints. The magnitude of the path derivatives i.e $x'(u), \alpha'(u)$ at the initial

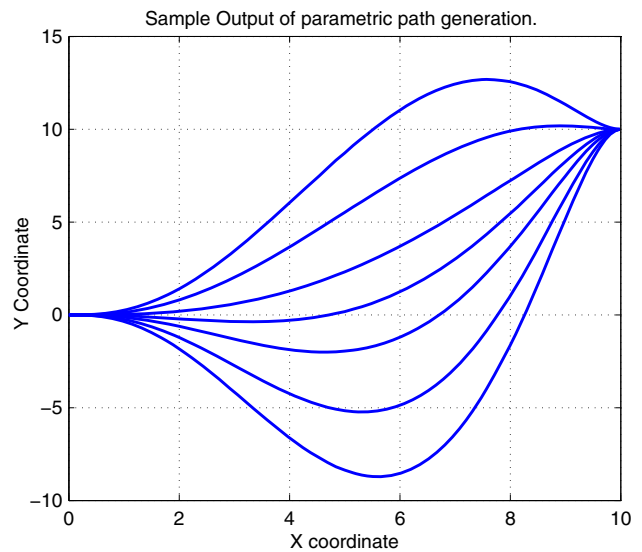


Fig. 3 Sample Output from the path generation framework showing how paths with various shapes can be generated by varying the intermediate points x_c, y_c

condition are kept equal to the $v_x, \dot{\alpha}$, the current x component linear velocity and rate of change of yaw angle of the robot. x_c, y_c represents an intermediate point on the path which is varied to control the shape of the path.

Figure 3 shows a sample output of the path generation framework. The various paths between the same start and the goal states are generated by varying the intermediate point x_c, y_c .

The yaw plane path obtained is mapped through some suitable functions to obtain a complete 6 dof path for all the states of the robot. The next section describes the derivation of these functions using the geometrical constraints of the robot and the terrain geometry.

3 Derivation of robot’s 6 dof states and dynamics

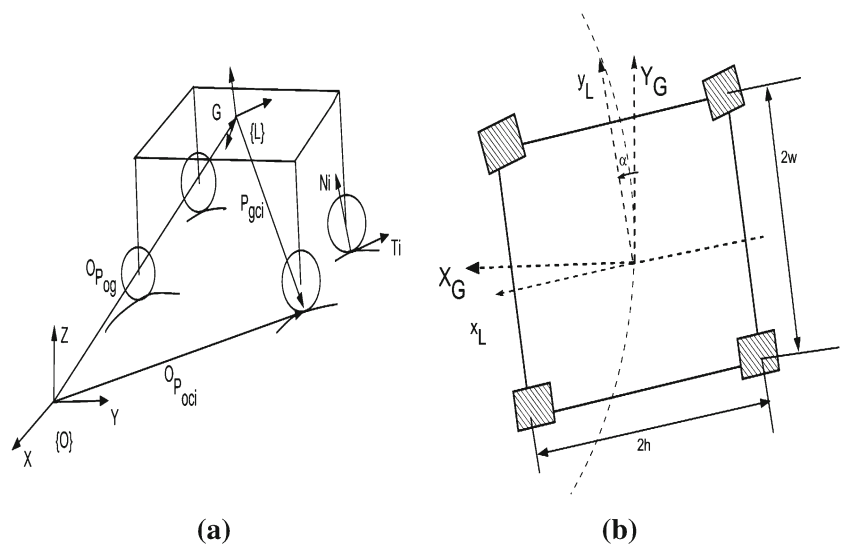
3.1 Robot’s 6 dof evolution

For mapping the yaw plane path to the full 6 dof path, we assume that the terrain equation can be known in the following form

$$a = f(b, c) \tag{11}$$

Where ‘a’ represents the terrain height at the x and y coordinate (b, c) . With the help of terrain equation this section derives analytical functions relating x, y, α to robot’s z coordinate, roll β and pitch γ . Along with it, x, y, α are also related to the twelve wheel ground contact points x_{ci}, y_{ci}, z_{ci} . These relations comes from the holonomic constraints defining the geometry of the robot (refer Fig. 4a)

Fig. 4 **a** A four wheeled robot with the vectors describing the holonomic constraint defining the geometry of the robot shown. **b** Top-view of the robot



$$\vec{P}_{og} + \vec{P}_{gci} = \vec{P}_{oci} \tag{12}$$

where

$$\begin{aligned} \vec{P}_{gci} &= \vec{r}_f \\ \vec{r}_f &= R \left[\delta h \frac{2.5-i}{|(2.5-i)|} w - (l_i) \right], \forall i = 1, 2, 3, 4 \\ \Delta = 1, i = 1, 4 \\ &= -1, i = 2, 3 \end{aligned}$$

R is the rotation matrix describing the orientation of the $\{G\}$ with respect to body fixed frame $\{L\}$. We assume the fixed angle representation for R . The frame $\{G\}$ has the same orientation as the inertial frame $\{0\}$ but moves along with the robot and is attached at the same point as frame $\{L\}$. $(2.5 - i)$ and δ has been incorporated to ensure proper sign of w and h corresponding to each vertex of the chassis. l_i are the equivalent leg lengths which includes the radius of the wheels. h and w are half width and breadth of the chassis.

We linearize (12) with respect to β and γ for each wheel as:

$$x_{ci} = x - \frac{2.5-i}{|(2.5-i)|} w \sin \alpha - l_i \gamma \sin \alpha + \delta h \cos \alpha - l_i \beta \cos \alpha \tag{13}$$

$$y_{ci} = y + \frac{2.5-i}{|(2.5-i)|} w \cos \alpha - l_i \beta \sin \alpha + \delta h \sin \alpha + l_i \gamma \cos \alpha \tag{14}$$

$$z_{ci} = z + \frac{2.5-i}{|(2.5-i)|} w \gamma - l_i - \delta h \beta \tag{15}$$

The variables x_{ci}, y_{ci}, z_{ci} satisfy (11) and to explicitly solve for β and γ as a function of x, y, α it is required that (11) could be represented as a combination of piecewise linear hyperplanes. In case when the fitted terrain equation to the

DEM is non-linear we can linearize the terrain equation about the vehicle’s geometric center. This linearisation is justified since any terrain can be locally represented by a linear plane having a particular orientation in 3D space. Linearizing (11) about the current chassis center coordinate gives

$$z_{ci} = k_3 + k_1(x_{ci} - x) + k_2(y_{ci} - y) \tag{16}$$

where $k_3 = f(x, y), k_1 = \frac{\partial(f)}{\partial b}, b = x, c = y, k_2 = \frac{\partial(f)}{\partial c}, b = x, c = y$.

Substituting x_{ci}, y_{ci}, z_{ci} values from (13), (14), and (15), four equations represented by (16) can be written in the matrix form as

$$\begin{bmatrix} 1 & w + \eta_1 & h + v_1 \\ 1 & w + \eta_2 & -h + v_2 \\ 1 & -w + \eta_3 & -h + v_3 \\ 1 & -w + \eta_4 & h + v_4 \end{bmatrix} \begin{bmatrix} z \\ \gamma \\ \beta \end{bmatrix} = \begin{bmatrix} H_1 \\ H_2 \\ H_3 \\ H_4 \end{bmatrix} \tag{17}$$

The terms η_i, v_i, H_i are defined in the Appendix 1 and are a function of the robots suspension travel length. The coefficient matrix in (17) can be pseudo-inverted to solve for z, β, γ . However, if the suspension travel length is small which essentially means that $\eta_1 = \eta_2 = \eta_3 = \eta_4 = \eta$, and $v_1 = v_2 = v_3 = v_4 = v$, with small matrix manipulation (17) can be reduced to

$$\begin{bmatrix} 0 & 2w & 0 \\ 0 & 0 & -2h \\ 1 & -w + \eta & h + v \end{bmatrix} \begin{bmatrix} z \\ \gamma \\ \beta \end{bmatrix} = \begin{bmatrix} H_2 - H_3 \\ H_3 - H_4 \\ H_4 \end{bmatrix} \tag{18}$$

Inverting coefficient matrix in (18) gives z, γ, β as analytical functions of x, y, α .

3.2 Approximate closed form versus exact numerical solution for posture determination

Determination of robot’s 3D posture and contact points is very important since it forms the basis for evaluating the stability constraints and the range of feasible speeds and accelerations discussed later in the paper. What is noteworthy and unique is how we relate the robot’s 3D posture and wheel ground contact points through the geometric holonomic constraints (Eq. 12) and surface equation. The derivation presented in the above section provides approximate closed form solutions to the resulting set of coupled non-linear equations. A more accurate and computationally involved methodology was used in our earlier work (Singh et al. 2011) where an exact solution to the non-linear equations were obtained through numerical means. For any given path, one could resort to the use of either approximate closed form or the more involved exact numerical solutions depending on the unevenness of the underlying terrain. This trade-off would prove beneficial from the computational standpoint. For example, consider Figs. 5a and 6a which shows robot’s

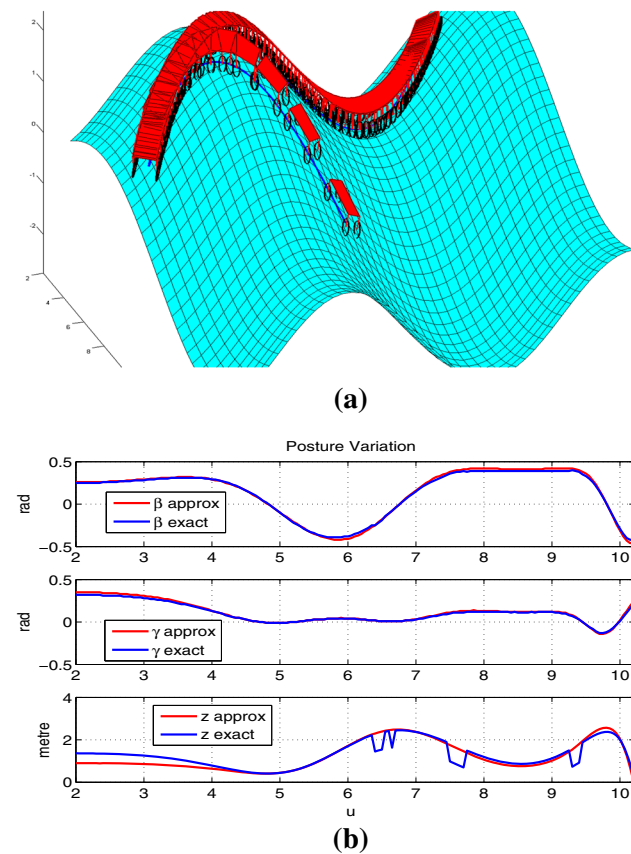


Fig. 5 **a** Robot evolution on uneven terrain. **b** Posture Variation along the path. The figure shows that the posture obtained through approximate analytical functions agree well with that obtained from solving (12) exactly in the non-linear form (Color figure online)

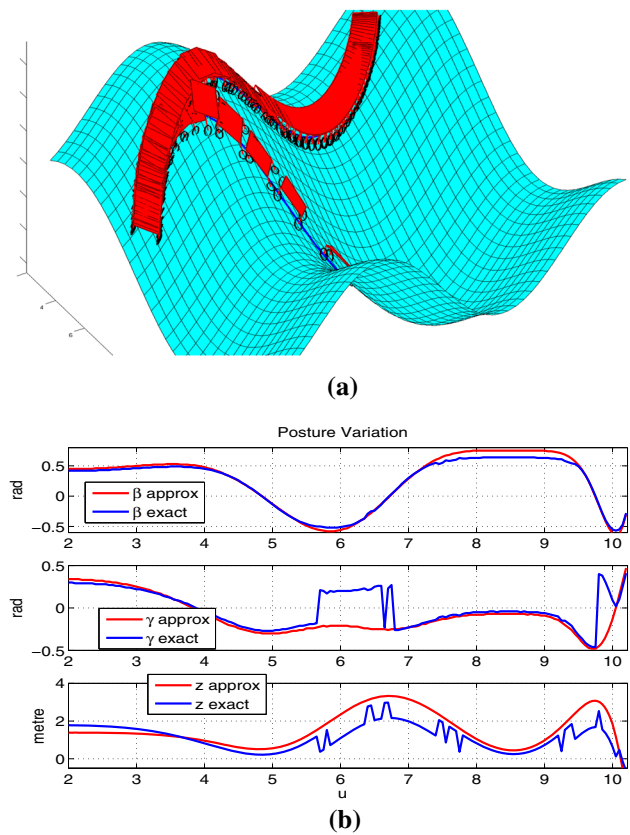


Fig. 6 **a** Robot’s evolution on uneven terrain. **b** The presented terrain is more uneven as compared to that shown in Fig. 5a. Although this increase in unevenness leads to deterioration in the accuracy of the approximate closed form, it still close to the exact numerical solution along most segments of the path (Color figure online)

evolution along a path on two different terrains. The terrain equations are respectively modelled as $(z = 1.05 \cos(0.4x) + 1.05 \sin(0.3y))$ and $(z = 1.4 \cos(0.4x) + 1.4 \sin(0.4y))$. Thus, the terrain shown in Fig. 6a is more uneven as it has higher amplitude and frequency of undulations. A comparison between the approximate closed form and the exact numerical solution on both the terrains is shown in Figs. 5b and 6b. It can be seen that there is a good agreement between the approximate closed form and the exact numerical solution for most segments of both the path. However, with the increase in unevenness, there is a gradual deterioration in the accuracy of the approximate solution. In general, we have observed in simulations that the agreement between the approximate and the exact model deteriorates with the increase in the frequency and amplitude of the terrain undulations. It should be noted that use of either approximate closed form or exact numerical solution does not in any way affect the structure of the framework presented in this paper. For example, in Sect. 4 only numerical values of the robot’s posture and it’s derivatives are required for evaluation of the stability constraints, which can be computed either through

approximate closed form models or by solving the geometric holonomic constraints and surface equations exactly through numerical means.

However, to exploit the computational advantage provided by the approximate closed form evolution model, it is imperative to know a priori the regions where there is good agreement between the approximate closed form and exact numerical solution. One approach to accomplish this would be to perform a one time off-line sampling of the given uneven terrain to divide it into two regions: one where numerical solution is required and one where approximate solution would suffice. Then depending on the queried start and goal state and on how much of a given path falls into which region, we could find a trade-off between opting for exact numerical and approximate closed form solution. A more rigorous analysis of the relation between the unevenness of terrain and accuracy of the approximate closed form solution is part of our future work.

3.3 Robot dynamics

The full 3D dynamics of the robot at any given state can be ascertained by exploiting the derivations described in the previous section. In particular, it can be noted that (13–15) allow us to have wheel ground contact points also as a function of x, y and α . Wheel ground contact points location are important because they decide contact surface unit normals $\hat{n}_i = [n_{xi}, n_{yi}, n_{zi}]^T$ which in turn decides the traction force unit vectors. Following the derivation mentioned in Appendix 3 the equations of motion of the vehicle can be presented in the following form

$$A * C = D \tag{19}$$

where $C = [T_i \ N_i]_{2n \times 1}^T$ $D = [m \vec{a} \ I \vec{\Omega} + \vec{\Omega} \times I \vec{\Omega}]_{6 \times 1}^T$ $T_i, N_i, \vec{a}, \vec{\Omega}, \vec{\Omega}$ are the traction, normal forces, linear and angular acceleration and angular velocity respectively. n is the number of wheels of the vehicle, m is the mass of the vehicle and $I_{3 \times 3}$ is the inertia matrix.

The elements of $A_{6 \times 2n}$ matrix depends on vehicle posture, geometry and terrain dependent parameters like surface contact normals and traction unit vectors. It is to be noted that because of the derivation presented from (13–18) matrix A can also be known in closed form as a function of x, y, α . Ideally, if matrix A could be inverted symbolically, we could have analytical functional description of the variation of T_i, N_i with respect to $\vec{a}, \vec{\Omega}$ and in theory we could have a gradient descent based algorithm to generate a one shot stable trajectory. However, robots operating on uneven terrain generally have 4-8 wheels which makes A under-constrained and have to be pseudo-inverted. Some algorithms like Jones

et al. (1996) computes symbolic pseudo inverse for small matrices having one or two independent variables. But, symbolically pseudo-inverting matrix A , whose dimension will increase with the number of wheels can turn into a complex problem. It should be noted that this problem is not unique to the framework proposed in this paper but is fundamental with modeling robot dynamics in 3D and relating traction and normal forces to velocity and acceleration in closed form. Hindered by this critical problem, a two step process of trajectory generation stated earlier in Sect. 1 is followed which utilizes the fact that it is relatively easy to compute the pseudo inverse numerically at any point on the terrain and by doing so we get the following equations.

$$T_i = a_{i1}(ma_x) + a_{i2}(ma_y) + a_{i3}(mg + ma_z) + a_{i4}(I_{xx}\dot{\Omega}_x + I_{zz}\Omega_y\Omega_z - I_{yy}\Omega_y\Omega_z) + a_{i5}(I_{yy}\dot{\Omega}_y + I_{xx}\Omega_x\Omega_z - I_{zz}\Omega_x\Omega_z) + a_{i6}(I_{zz}\dot{\Omega}_z + I_{yy}\Omega_x\Omega_y - I_{xx}\Omega_x\Omega_y) \tag{20}$$

$$N_i = a_{j1}(ma_x) + a_{j2}(ma_y) + a_{j3}(mg + ma_z) + a_{j4}(I_{xx}\dot{\Omega}_x + I_{zz}\Omega_y\Omega_z - I_{yy}\Omega_y\Omega_z) + a_{j5}(I_{yy}\dot{\Omega}_y + I_{xx}\Omega_x\Omega_z - I_{zz}\Omega_x\Omega_z) + a_{j6}(I_{zz}\dot{\Omega}_z + I_{yy}\Omega_x\Omega_y - I_{xx}\Omega_x\Omega_y) \tag{21}$$

$\forall i = \{1, 2, 3, 4\}, \forall j = \{5, 6, 7, 8\}$. g is acceleration due to gravity. The coefficients $a_{i1}, a_{i3}, a_{j2}, a_{in}, a_{jn}$, are the elements of the pseudo inverse of matrix A . A feasible combination of linear and angular acceleration is determined by the elements of the pseudo-inverse matrix and is defined as one which satisfies the following constraints.

$$N_i > 0 \tag{22}$$

$$|(T_i)| < \rho N_i \Rightarrow T_i > -\rho N_i, T_i < \rho N_i \tag{23}$$

The term ρ represents the coefficient of friction in (23). It can be inferred from (20) and (21) that configuration x, y, α and terrain conditions which lead to $a_{j3} > 0$ are promising candidate for satisfying (22) and (23) since these coefficients decides the contribution of the weight. The contribution of the weight will always help in satisfaction of (22) as long as $a_{j3} > 0$. These coefficients can be rapidly evaluated for some look ahead distance along various directions and this information is fed to the path generation framework described in the Sect. 2 to control the shape of the path between start and a goal location.

4 Existence/non-existence of stable motion profile

In this section we describe a framework to quickly ascertain whether a stable motion profile could at all exist for a given candidate path. The previous two sections in combi-

nation provide a framework for computing $x(u)$, $y(u)$, $\alpha(u)$, $z(u)$, $\beta(u)$, $\gamma(u)$ and their derivatives. As shown in (4), to obtain the time dependent trajectory, a scaling transformation has to be performed for which the scaling function \dot{u} needs to be computed. The scaling function should be such that it results in velocities and accelerations that satisfy (22) and (23). Thus, we first determine the range of values that the scaling function \dot{u} is allowed to take. To this end let s_u denote the scale factor by which the path derivatives needs to be scaled to satisfy (22) and (23). Thus, we can write

$$\dot{x}(t) = x'(u)s_u, \ddot{x}(t) = x''(u)s_u^2 \tag{24}$$

$$\dot{y}(t) = y'(u)s_u, \ddot{y}(t) = y''(u)s_u^2 \tag{25}$$

$$\dot{\alpha}(t) = \alpha'(u)s_u, \ddot{\alpha}(t) = \alpha''(u)s_u^2 \tag{26}$$

Through (24), (25) and (26) and following the definition given in Appendix 2, we can infer

$$\vec{\dot{z}}(t) = \vec{\dot{z}}'(u)s_u^2, \ddot{z}(t) = z''(u)s_u^2 \tag{27}$$

Keeping in mind the fact that the scaling transformation does not change the path of the function, at every point along the obtained path $\mathbf{X}(u)$, (24), (25), (26) and (27) are substituted in (22) and (23) to get equations in the following form.

$$s_u^2 C_{Ni} > D_{Ni} \tag{28}$$

$$s_u^2 C_{Tai} < D_{Tai}, s_u^2 C_{Tbi} < D_{Tbi} \tag{29}$$

Here i represents the i^{th} wheel and $i = 1, 2, 3, 4$ and $j = 5, 6, 7, 8$. Also $j = 5$ when $i = 1, j = 6$ when $i = 2$ and so on. The coefficients are given below as

$$C_{Ni} = a_{j1}(mx''(u)) + a_{j2}(my''(u)) + a_{j3}(mz''(u))$$

$$+ a_{j4}(I_{xx}\Omega'_x(u) + I_{zz}\Omega_y(u)\Omega_z(u) - I_{yy}\Omega_y(u)\Omega_z(u))$$

$$+ a_{j5}(I_{yy}\Omega'_y(u) + I_{xx}\Omega_x(u)\Omega_z(u) - I_{zz}\Omega_x(u)\Omega_z(u))$$

$$+ a_{j6}(I_{zz}\Omega'_z(u) + I_{yy}\Omega_x(u)\Omega_y(u) - I_{xx}\Omega_x(u)\Omega_y(u))$$

$$D_{Ni} = -a_{j3}mg$$

$$C_{Tai} = -(\rho a_{j1} + a_{i1})mx''(u) - (\rho a_{j2} + a_{i2})(my''(u))$$

$$- (\rho a_{j3} + a_{i3})(mz''(u)) - (\rho a_{j4} + a_{i4})(I_{xx}\Omega'_x(u))$$

$$+ I_{zz}\Omega_y(u)\Omega_z(u) - I_{yy}\Omega_y(u)\Omega_z(u))$$

$$- (\rho a_{j5} + a_{i5})(I_{yy}\Omega'_y(u))$$

$$+ I_{xx}\Omega_x(u)\Omega_z(u) - I_{zz}\Omega_x(u)\Omega_z(u))(\rho a_{j6}$$

$$+ a_{i6})(I_{zz}\Omega'_z(u))$$

$$+ I_{yy}\Omega_x(u)\Omega_y(u) - I_{xx}\Omega_x(u)\Omega_y(u))$$

$$D_{Tai} = a_{i3}mg + \rho a_{j3}mg$$

$$C_{Tbi} = (-\rho a_{j1} + a_{i1})mx''(u)$$

$$+ (-\rho a_{j2} + a_{i2})(my''(u)) + (-\rho a_{j3} + a_{i3})(mz''(u))$$

$$+ (-\rho a_{j4} + a_{i4})(I_{xx}\Omega'_x(u) + I_{zz}\Omega_y(u)\Omega_z(u))$$

$$- I_{yy}\Omega_y(u)\Omega_z(u))$$

$$+ (-\rho a_{j5} + a_{i5})(I_{yy}\Omega'_y(u) + I_{xx}\Omega_x(u)\Omega_z(u))$$

$$- I_{zz}\Omega_x(u)\Omega_z(u))$$

$$+ (-\rho a_{j6} + a_{i6})(I_{zz}\Omega'_z(u) + I_{yy}\Omega_x(u)\Omega_y(u))$$

$$- I_{xx}\Omega_x(u)\Omega_y(u))$$

$$D_{Tbi} = \rho a_{j3}mg - a_{i3}mg$$

Equations (28) and (29) along with $s_u > 0$ written for each wheel on any particular point on the path represents quadratic inequalities, the solution of which is required to construct the scaling function. The quadratic inequalities can be easily decomposed into two linear inequalities.

If no solution exists to the system of inequalities, it can be concluded that no stable motion profile exists for the given parametric curve representing the path. It can be seen that the solution to the system of inequalities depends upon the coefficients of the pseudo inverse of the matrix A i.e a_{i1} , a_{j1} which in turn depends on vehicle geometry, terrain conditions and vehicle posture. To have an insight on how the solution space of s_u is computed, consider (28), it's coefficients C_{Ni} , D_{Ni} and the following four cases:

• Case 1

If for a particular wheel $C_{Ni} > 0$, $D_{Ni} > 0$, then $\kappa = \frac{C_{Ni}}{D_{Ni}} > 0$ and we have

$$s_u^2 > \kappa \tag{30}$$

• Case 2

If for a particular wheel $C_{Ni} < 0$, $D_{Ni} > 0$, then $\kappa = \frac{C_{Ni}}{D_{Ni}} < 0$ and we have

$$s_u^2 < \kappa \tag{31}$$

It can be seen that no real value satisfies (31) and hence no solution for s_u exists if for any particular wheel, this case arises

• Case 3

If for a particular wheel $C_{Ni} > 0$, $D_{Ni} < 0$, then $\kappa = \frac{C_{Ni}}{D_{Ni}} < 0$ and we have

$$s_u^2 > \kappa \tag{32}$$

It can be seen that any non-zero real value will satisfy (32)

• Case 4

If for a particular wheel $C_{Ni} < 0$, $D_{Ni} < 0$, then $\kappa = \frac{C_{Ni}}{D_{Ni}} > 0$ and we have

$$s_u^2 < \kappa \tag{33}$$

For any of the wheel either of the four cases could occur. The solution space of s_u will be an intersection of the individual solution spaces of each wheel. The solution space of s_u will be further shortened when the no-slip constraints are enforced. By factoring the quadratic inequalities into two linear inequalities, a linear programming based approach can be followed for computing the maximum and minimum value of the solution space of s_u . Alternatively, it is also possible to adopt a search based method to compute the range of s_u . A pre-defined set of values for s_u can be generated and checked whether they lead to satisfaction of (28) and (29)

Besides providing the solution space for s_u , the inequalities (30–33) serves a deeper purpose. It provides analytical formulae for checking whether a given path is stable or not. For, example if for any point on the given path, for any of the wheel, the constraint is in the form (31), then no stable velocity or acceleration profiles exists and that particular wheel will lift off the ground while traversing the path. Moreover, the procedure for checking can be very fast since the coefficients C_{Ni} and D_{Ni} are known in semi-closed form in terms of path parameters, vehicle posture and terrain geometry.

5 Construction of scaling functions

In this section we describe a fast and simple framework to obtain a feasible scaling function. A feasible scaling function is one which assumes values from the solution space of s_u so that the resulting velocity and acceleration profile satisfy (22) and (23). Let the maximum and minimum value of the solution space of s_u , at each point be denoted as $s_{max}(u)$ and $s_{min}(u)$. Then using (4), we can say that the scaling function should satisfy the following inequalities.

$$s_{min}(u) \leq \frac{du}{dt} \leq s_{max}(u) \tag{34}$$

$$|x''(u)|s_{min}^2(u) \leq |x''(u)(\dot{u})^2 + x'(u)\ddot{u}| \leq |x''(u)|s_{max}^2(u) \tag{35}$$

$$|y''(u)|s_{min}^2(u) \leq |y''(u)(\dot{u})^2 + y'(u)\ddot{u}| \leq |y''(u)|s_{max}^2(u) \tag{36}$$

$$|\alpha''(u)|s_{min}^2(u) \leq |\alpha''(u)(\ddot{u})^2 + \alpha'(u)\dot{u}| \leq |\alpha''(u)|s_{max}^2(u) \tag{37}$$

$$sign(x''(u)(\dot{u})^2 + x'(u)\ddot{u}) = sign(x''(u)), \forall u \in [u_0, u_f] \tag{38}$$

$$sign(y''(u)(\dot{u})^2 + y'(u)\ddot{u}) = sign(y''(u)), \forall u \in [u_0, u_f] \tag{39}$$

$$sign(\alpha''(u)(\dot{u})^2 + \alpha'(u)\ddot{u}) = sign(\alpha''(u)), \forall u \in [u_0, u_f] \tag{40}$$

The physical interpretation of (34–40) is that we have to create a scaling function which remains bounded by s_{min} and s_{max} curve while satisfying the acceleration level inequalities.

The scaling function \dot{u} is obtained as a combination of various exponential functions of the form given in (6). To understand this further consider Fig. 7 where the path variable interval $[u_0, u_f]$ is divided into $n + 1$ subintervals by the grid points $u_1, u_2, u_3, \dots, u_n$. In each subinterval, i.e between any two grid points, a different exponential function determined by parameters $p_1, q_1, p_2, q_2, \dots, p_n, q_n, p_{n+1}, q_{n+1}$ is defined. The scaling function \dot{u} is a function of these parameters, which can be obtained by solving the inequalities (34–37) at the grid points. Before we describe the solution procedure, we simplify (34–37) and express them as explicit functions of p_i and q_i .

5.1 Simplification of inequalities (34–40)

The simplification of inequalities (34–40) depends on obtaining a general form of \dot{u} at an arbitrary grid point in terms of parameters p_i and q_i . To this end, we start with the first subinterval $[u_0, u_1]$. Let the initial boundary value of \dot{u} be s_1 . To satisfy the initial boundary condition, we must have

$$\dot{u}(u_0) = p_1 e^{-q_1 u_0} = s_1 \Rightarrow p_1 = s_1 e^{q_1 u_0} \tag{41}$$

Using (41) \dot{u} in the interval $[u_0, u_1]$ can be represented in the following form

$$\dot{u}(u) = s_1 e^{q_1(u_0 - u)}, \forall u \in [u_0, u_1] \tag{42}$$

Similarly from Fig. 7, it can be seen that in the interval $[u_1, u_2]$, \dot{u} is defined as $p_2 e^{-q_2 u}$. To ensure continuity with the scaling function of the preceding interval we must have

$$\begin{aligned} \dot{u}(u_1) = p_2 e^{-q_2 u_1} &\Rightarrow s_1 e^{q_1(u_0 - u_1)} = p_2 e^{-q_2 u_1} \\ &\Rightarrow p_2 = s_1 e^{q_1(u_0 - u_1) + q_2 u_1} \end{aligned} \tag{43}$$

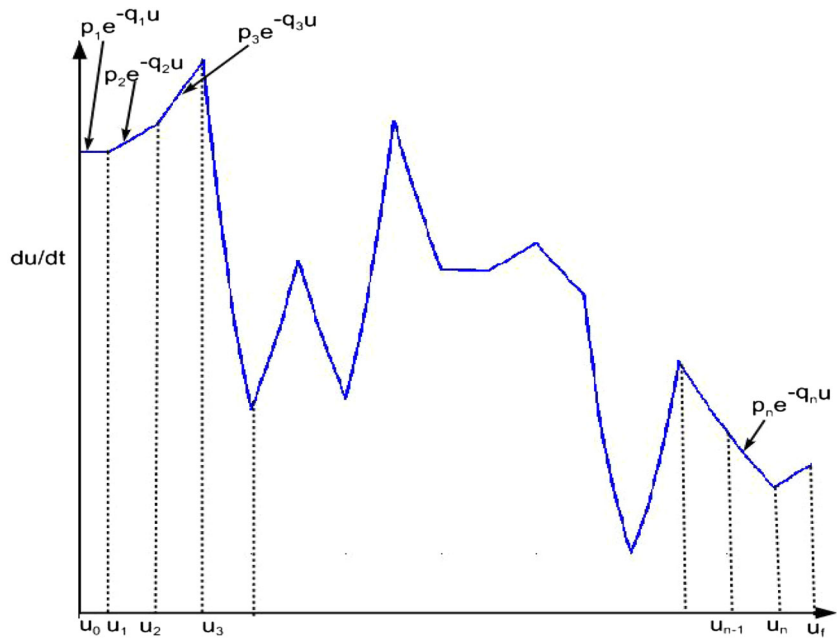
Using (43), \dot{u} in the interval $[u_1, u_2]$ can be defined as

$$\dot{u}(u) = s_1 e^{q_1(u_0 - u_1) + q_2(u_1 - u)}, \forall u \in [u_1, u_2] \tag{44}$$

Following the same procedure as above \dot{u} at the n^{th} subinterval i.e between grid points u_{n-1} and u_n can be represented as

$$\dot{u}(u) = s_1 e^{q_1(u_0 - u_1) + q_2(u_1 - u_2) + \dots + q_n(u_{n-1} - u)}, \forall u \in [u_{n-1}, u_n] \tag{45}$$

Fig. 7 Discretization of path variable interval into various sub-intervals by the grid points $u_1, u_2, u_3 \dots u_n$. An exponential function is fitted in each subinterval. The scaling function \dot{u} is obtained as a continuous combination of the exponential functions



\dot{u} at any arbitrary grid point can be represented using (45) as

$$\dot{u}(u_n) = s_1 e^{q_1(u_0-u_1)+\dots+q_n(u_{n-1}-u_n)} \tag{46}$$

Following the derivations above, it is straightforward to note that \ddot{u} at the n^{th} subinterval can be written as

$$\begin{aligned} \ddot{u}(u) &= -q_n e^{2q_1(u_0-u_1)+\dots+2q_n(u_{n-1}-u)} , \forall u \in [u_{n-1}, u_n] \\ &= -q_n \dot{u}^2(u) \end{aligned} \tag{47}$$

Hence \ddot{u} at any arbitrary grid point can be written as

$$\ddot{u}(u_n) = -q_n \dot{u}^2(u_n) \tag{48}$$

Now evaluating inequality (34) at the grid points and taking the logarithmic transformation results in following $2(n + 1)$ linear inequalities

$$C_{vi} = \begin{cases} \log(\dot{u}_i) - \log(s_{max}(u_i)) \leq 0 \\ -\log(\dot{u}_i) + \log(s_{min}(u_i)) \leq 0, \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{49}$$

Similarly (35–40) can be simplified as following

$$C_{axi} = \begin{cases} 2 \log(\dot{u}(u_i)) + \log(|x''(u_i) - q_i x'(u_i)|) \\ - \log(|x''_{max}(u_i)| s_{max}^2(u)) \leq 0 \\ -2 \log(\dot{u}(u_i)) - \log(|x''(u_i) - q_i x'(u_i)|) \\ + \log(|x''_{max}(u_i)| s_{min}^2(u)) \leq 0 \\ , \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{50}$$

$$C_{ayi} = \begin{cases} 2 \log(\dot{u}(u_i)) + \log(|y''(u_i) - q_i y'(u_i)|) - \\ \log(|y''_{max}(u_i)| s_{max}^2(u)) \leq 0 \\ -2 \log(\dot{u}(u_i)) - \log(|y''(u_i) - q_i y'(u_i)|) \\ + \log(|y''_{max}(u_i)| s_{min}^2(u)) \leq 0 \\ , \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{51}$$

$$C_{axi} = \begin{cases} 2 \log(\dot{u}(u_i)) + \log(|\alpha''(u_i) - q_i \alpha'(u_i)|) \\ - \log(|\alpha''_{max}(u_i)| s_{max}^2(u)) \leq 0 \\ -2 \log(\dot{u}(u_i)) - \log(|\alpha''(u_i) - q_i \alpha'(u_i)|) \\ + \log(|\alpha''_{max}(u_i)| s_{min}^2(u)) \leq 0 \\ , \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{52}$$

$$C_{six} = \begin{cases} -x''(u) + q_i x'(u) < 0, x''(u) > 0 \\ x''(u) - q_i x'(u) < 0, x''(u) < 0 \\ , \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{53}$$

$$C_{siy} = \begin{cases} -y''(u) + q_i y'(u) < 0, y''(u) > 0 \\ y''(u) - q_i y'(u) < 0, y''(u) < 0 \\ , \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{54}$$

$$C_{sia} = \begin{cases} -\alpha''(u) + q_i \alpha'(u) < 0, \alpha''(u) > 0 \\ \alpha''(u) - q_i \alpha'(u) < 0, \alpha''(u) < 0 \\ , \forall i = 1, 2, 3 \dots n, f \end{cases} \tag{55}$$

5.2 Solving (49–55)

To solve the inequalities (49–55) for the parameters q_i (p_i has already been eliminated), we fit them as constraints in the following non-linear optimization problem.

$$\begin{aligned} \min J &= \sum_{i=1}^{i=n} [\log(\dot{u}(u_i)) - \log(s_{max}(u_i))]^2 \\ s.t. & C_{vi} \leq 0, C_{axi} \leq 0, C_{ayi} \leq 0, C_{axi} \leq 0, \\ & C_{six} \leq 0, C_{siy} \leq 0, C_{sia} \leq 0 \end{aligned} \tag{56}$$

The optimization problem (56) has a fairly simple structure. The constraints C_{vi} are linear in terms of parameters q_i , since logarithm of an exponential function is linear. Similarly, the constraints $C_{s_{ix}}$, $C_{s_{iy}}$ and $C_{s_{i\alpha}}$ are linear in terms of the parameters q_i . The first term in the acceleration constraints (50–52) are linear. The second term in the acceleration constraints can be reformulated as a purely concave non-linearity, thus giving the constraints a *difference of convex* form (Singh 2015, Ch.5). The objective function is quadratic in terms of parameters q_i and seeks to bring the scaling function as close as possible to the s_{max} profile. This translates to higher velocities along the path which in turn optimizes time.

5.2.1 Initial guess for solving non-linear optimization (56)

A good initial guess of parameters q_i for solving (56) can be obtained by solving the following simpler optimization problem.

$$\begin{aligned} \min J = & \sum_{i=1}^{i=n} [\log(\dot{u}(u_i)) - \log(s_{max}(u_i))]^2 \\ \text{s.t. } & C_{vi} \leq 0 \end{aligned} \quad (57)$$

The optimization (57) is a convex quadratic programme and hence can be solved efficiently. Figure 8a, b shows the sample output obtained from solving optimization (56) for two different s_{max} and s_{min} profile.

6 Trajectory replanning

If for a particular path, no s_u exists at some points, or the scaling function satisfying (35–37) could not be created, then trajectory replanning is invoked to modify those portions of the trajectory. Let u_r correspond to the point from where the replanning needs to be done. From that point onwards the path generation problem represented by (10) is solved in the domain $u_r \leq u \leq u_f$ ensuring the continuity in the velocity and position space. Here the possible robot stability along various directions as measured by the variation of the coefficient a_{j3} described in Sect. 3 is used as a guiding factor to control the shape of the trajectory. We generate multiple paths passing through the regions having appropriate a_{j3} . The final feasible trajectory will be a concatenation of many continuously connected stable trajectories.

7 Simulation results

7.1 Example 1

The framework described above has been used here to generate trajectories on 3D uneven terrain. A small vehicle model with $m = 10\text{kg}$, $\rho = 0.7$ and with dimension

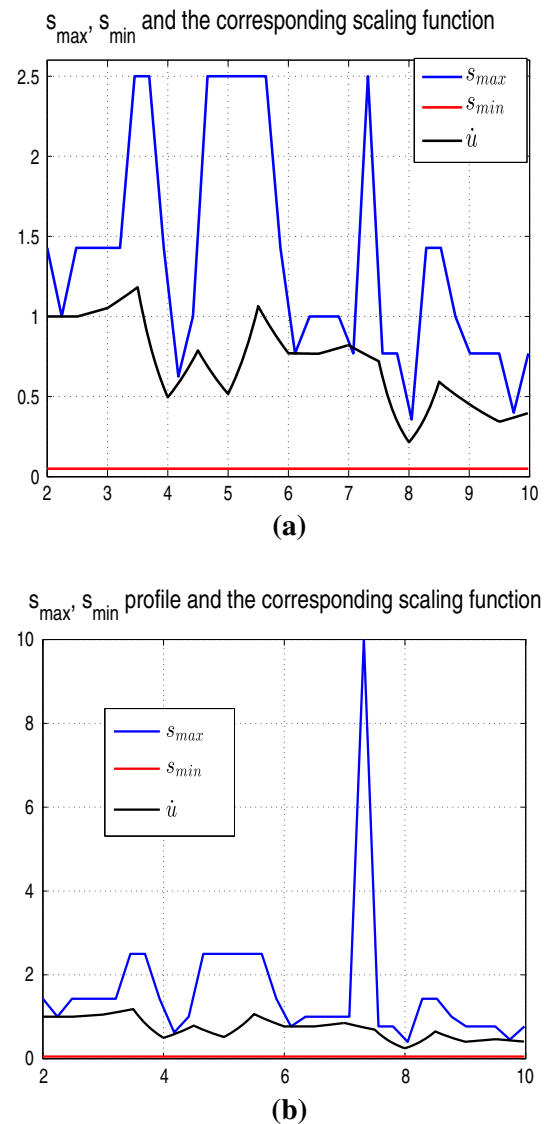


Fig. 8 a, b Scaling functions obtained through the non-linear optimization for the corresponding s_{max} and s_{min} profiles (Color figure online)

$1 \times 1 \times 1\text{m}^3$ was used in the simulation. The terrain is modelled as $z = 3.5(0.3\cos(0.3x) + 0.3\sin(0.3y))$.

An initial path $\chi_1(u)$ is produced by the planner as shown in Fig. 9a. The scale factor calculation described in Sect. 4 is performed over it. As shown in Fig. 9c, at various points along $\chi_1(u)$, scale factor does not exist and hence a stable motion profile could not be computed for it. The planner then searches for directions having $a_{j3} > 0$ and perturbs $\chi_1(u)$ along it. The second path, $\chi_2(u)$ produced by the planner is more stable than $\chi_1(u)$ as inferred from Fig. 9c which shows reduction in the number of points where scale factor does not exist. The stability of the third path $\chi_3(u)$ produced by further perturbation slightly reduces as compared to preceding path $\chi_2(u)$. Hence, the perturbation direction is adjusted resulting in paths $\chi_4(u)$ and $\chi_5(u)$. As can be seen from Fig. 9c that

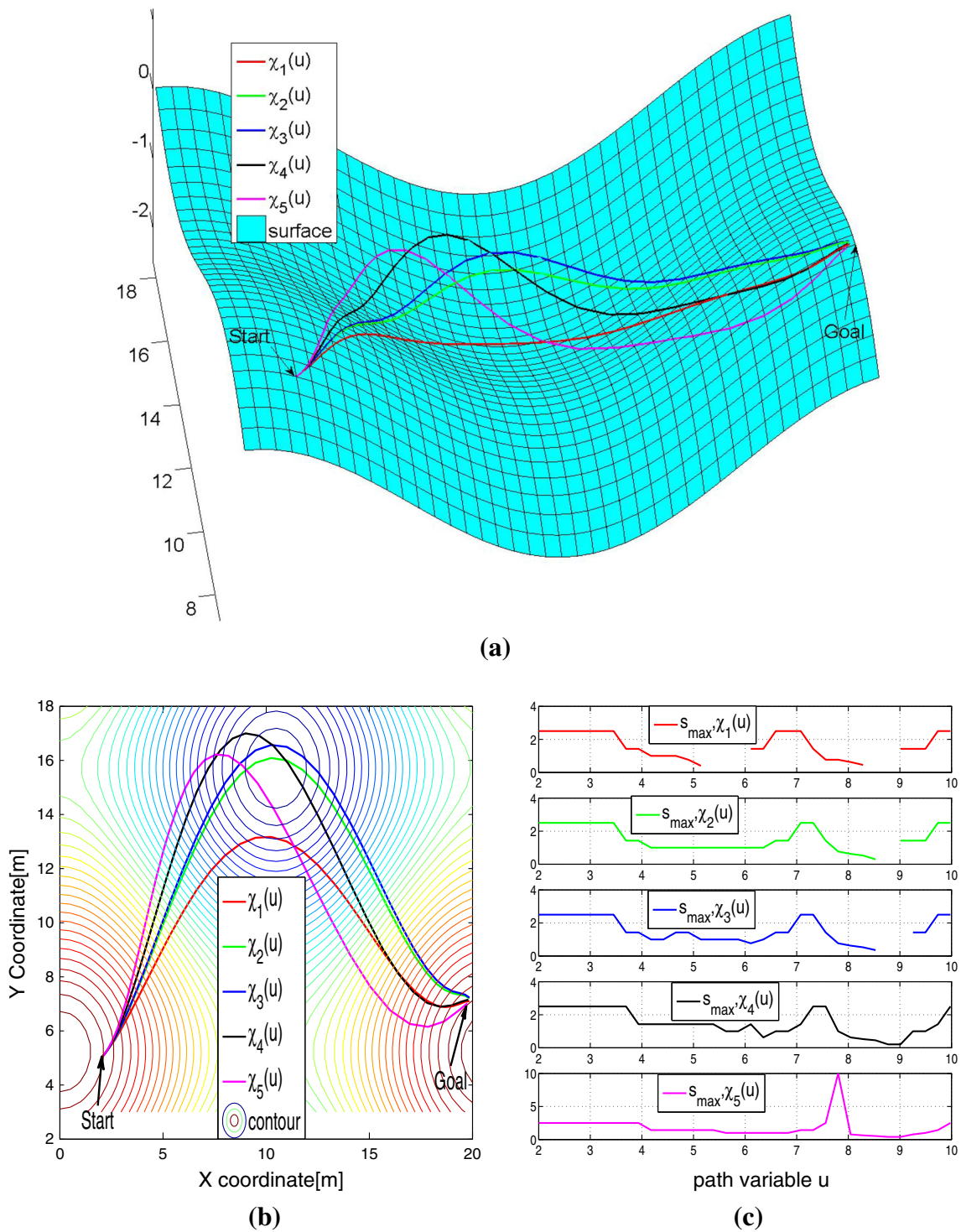


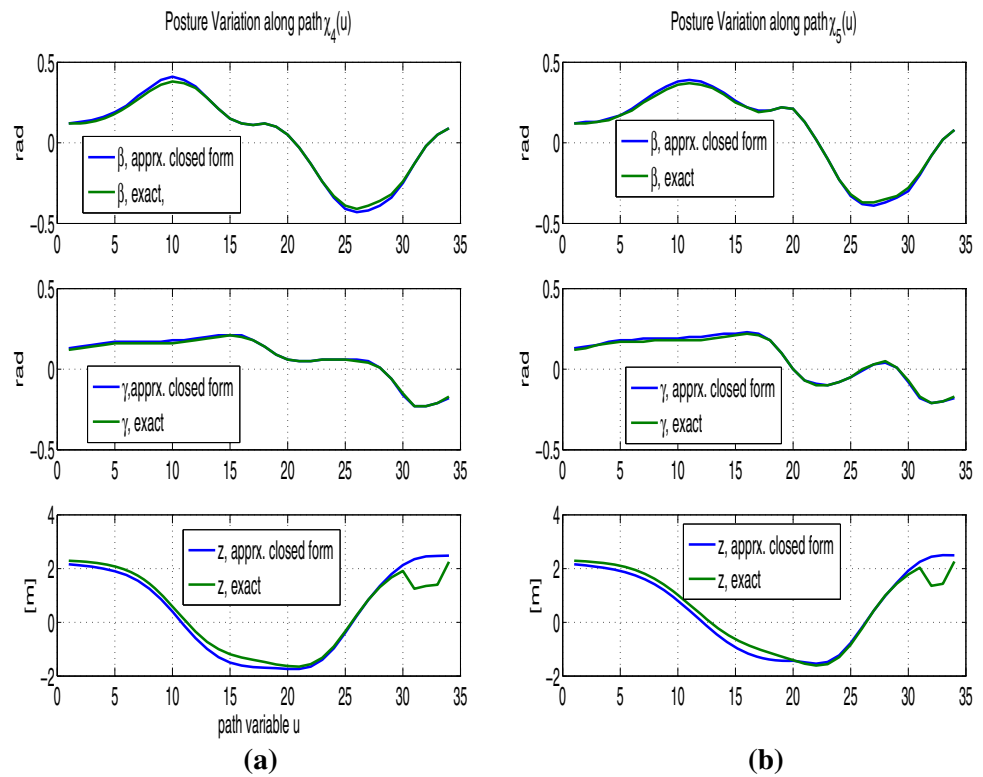
Fig. 9 **a** Paths planned between a given start and goal location. Various paths are planned and evaluated by the planner before arriving at the final completely stable path. **b** Yaw plane view of the paths. **c** Scale factor variation along the planned paths. It can be seen that the stability

of the paths subsequently increases with each perturbation. The number of points along the path at which the scale factor does not exist gives a measure of stability/instability of the path (Color figure online)

these two paths are completely stable as scale factor exists at each point along these paths. It can be seen from Fig. 9a that with each perturbation, the paths become more aligned with

gradient or slope of the terrain. Thus, gradual increase in the stability of the paths agree with the common intuition that it is easier to move along the slope rather than to cut across it.

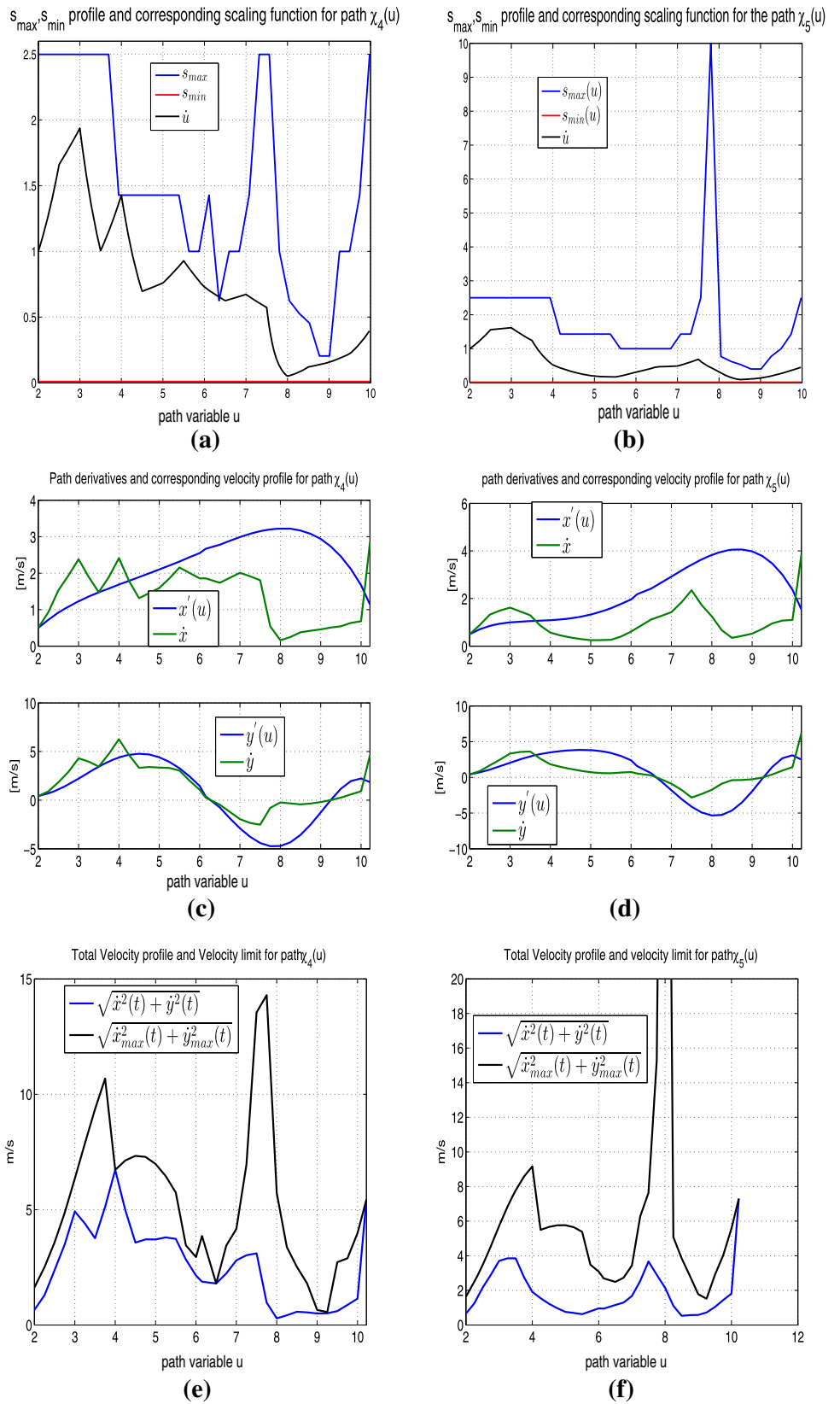
Fig. 10 **a, b** Posture variation along the stable paths $\chi_4(u)$ and $\chi_5(u)$. It can be again seen that the posture obtained through approximate closed form functions agree well with that obtained through solving (12) exactly in non-linear form. **c, d** Vehicle 3D evolution along the stable paths $\chi_4(u)$ and $\chi_5(u)$ (Color figure online)



The results of 6 *dof* state determination of the robot along the stable paths $\chi_4(u)$ and $\chi_5(u)$ are shown in Fig. 10a, b. It can be again be seen that the posture of the robot obtained

through approximate analytical expressions agree well with that obtained from solving non linear holonomic constraints exactly (Eq. 12) through numerical means.

Fig. 11 **a, b** Scaling function obtained through non-linear optimization (56) for the paths $\chi_4(u)$ and $\chi_5(u)$. As stated earlier, the quality of the scaling function is judged by how close it is to the s_{max} curve. Closer the scaling function is to the s_{max} curve, higher the resulting velocity along the given stable path. **c, d** Path derivatives and the corresponding velocity components obtained through scaling for the paths $\chi_4(u)$ and $\chi_5(u)$ respectively. **e, f** Total velocity profile and total velocity limit curves along path $\chi_4(u)$ and $\chi_5(u)$ respectively. Points where scaling function's magnitude is on the s_{max} curve corresponds to points where the total velocity profile is on the velocity limit curve (Color figure online)



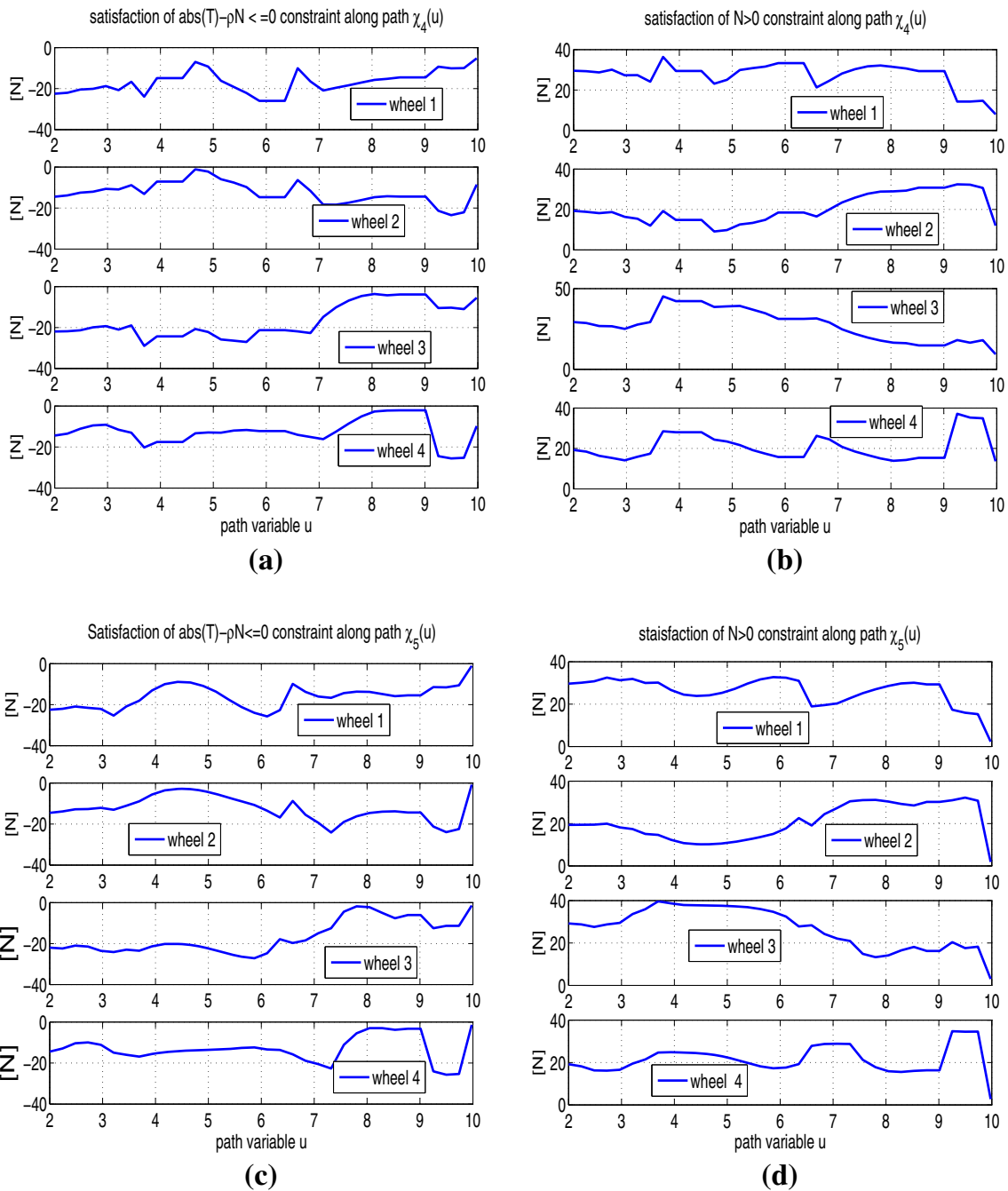


Fig. 12 a–d Satisfaction of no-slip and permanent contact constraints along the paths $\chi_4(u)$ and $\chi_5(u)$

The scaling function constructed for the scale factor profile obtained for paths $\chi_4(u)$ and $\chi_5(u)$ are shown in Fig. 11a, b. The quality of scaling function obtained for path $\chi_4(u)$ is better than that obtained for the path $\chi_5(u)$ as the former is more closer to the s_{max} profile. As stated earlier, closer the scaling function is to the s_{max} profile, higher the magnitude of the velocity profile along the path. The velocity profile obtained through scaling transformations are shown in Fig. 11c, d. Figure 11e, f shows the total velocity profile and the total velocity limit

is defined as 58. As can be observed, the points where total velocity profile is on the limit curve corresponds to the points where scaling function is on the s_{max} curve. Figure 12a–d shows the satisfaction of the no-slip and permanent wheel ground contact constraints along the obtained stable paths. It can be seen that the normal contact forces are always greater than zero, signifying permanent wheel ground contact at each point along the path. Further, the residual $|T_i| - \rho N_i$ is always negative, signifying the satisfaction of no-slip constraints.

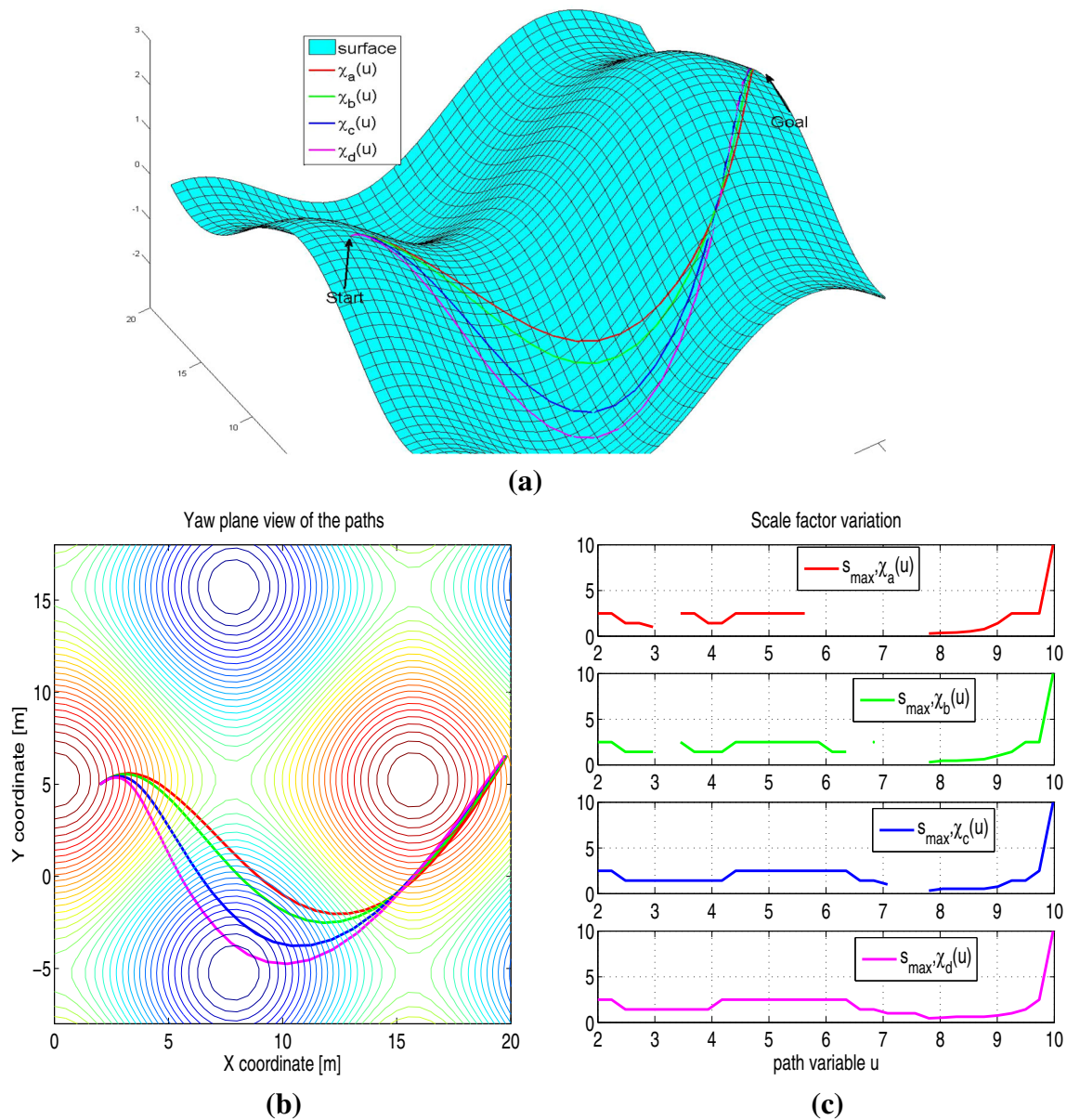


Fig. 13 **a** Effect of terrain complexity on trajectory generation. A slight increase in the frequency of the undulation of the terrain results in very different paths as compared to that obtained in example 1. The stable regions changes and hence accordingly the direction of perturbation

also changes as compared to example 1 (Fig. 9). **b** Yaw plane view of the paths. **c** Scale factor variation along the various paths evaluated by the planner (Color figure online)

$$V_{max} = \sqrt{\dot{x}_{max}^2(t) + \dot{y}_{max}^2(t)} = s_{max} \sqrt{\dot{x}(u)^2 + \dot{y}(u)^2} \quad (58)$$

7.2 Example 2: effect of terrain complexity on trajectory generation

This example shows how even slight changes in the terrain complexity significantly affects the geometry of the stable path and its corresponding motion profile. To this end, the planner was used between same start and goal state as example 1, but the terrain equation was now changed to $z = 3.5(0.3\cos(0.4x) + 0.3\sin(0.3y))$.

As can be seen, compared to previous example, the frequency of undulation has been slightly increased. However, as a result the stable regions as measured by the variation of coefficient a_{j3} significantly changes and hence the perturbation of the paths takes a very different form as compared to example 1. The sequence of perturbed paths generated by the planner is shown in Fig. 13a. As Similar to the previous example, the perturbation acts in a manner that each subsequent path is more stable than it’s preceding counterpart. This can be confirmed through Fig. 13c, which shows that the variation of the scale factor s_u along each computed path. The

number of points along the path for which the scale factor s_u does not exist reduces with each perturbation.

7.3 Example: trajectory re-planning

In situations where the planner produces paths which are stable for a considerable distance, then instead of searching for one shot-stable path, the planner resorts to re-planning and a new path is generated keeping the stable portions of the previous path intact. Consider Fig. 14 which shows a path $\chi(u)$ (in blue) having some unstable portions (shown in black). So keeping the stable portions intact a new stable path $\chi_r(u)$ is generated to the goal. In Fig. 14 only the final re-planned path is shown. Actually the planner like the previous example produces various intermediate re-planned paths passing through regions having $a_{j3} > 0$. $X - Y$ projection of the paths are shown in Fig. 15a. The scale factor variation along the initial path $\chi(u)$, the re-planned path $\chi_r(u)$ and the concatenated path $\chi(u) + \chi_r(u)$ is shown in Fig. 15b.

7.4 Comparison of trajectories with point mass dynamics and the proposed full 3D dynamics

Figure 16a shows the comparison between the paths obtained with a point mass dynamics and the proposed full dynamics. Even while obtaining the path with point mass dynamics, it was ensured that a valid stable posture of the vehicle exists along the path. To understand the significance of the proposed 3D model, we evolve the full vehicle model along the path obtained with point mass dynamics and compare the resulting scale factor obtained. This is shown in Fig. 16b. It can be seen that from the point of view of point mass dynamics, the path is stable but if the full 3D dynamics is considered certain points along the path does not satisfy the no-slip and

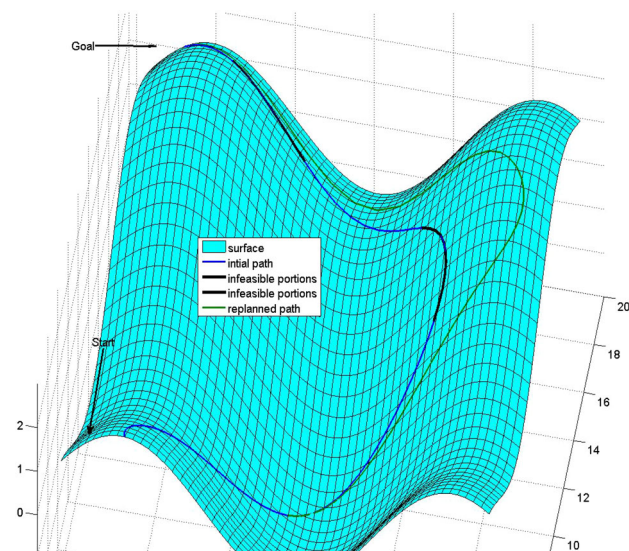


Fig. 14 A initial path $\chi(u)$ which is unstable beyond some point is re-planned while keeping the stable portions of the path intact

permanent contact condition and hence no scale factor exists on those points. This stark difference arises out of a subtle yet critical difference between the proposed work and Shiller's dynamic planning (2000) using point mass. Shiller's work (2000) considers the resultant traction and normal forces due to all wheels at the center of mass of the vehicle and then apply the no-slip and permanent contact constraint on the resultant forces. But even when the resultant forces satisfy constraints (22) and (23), each individual component arising out of each wheel ground interaction may or may not satisfy them. For example, the resultant normal force may be greater than zero even when any one or more wheel ground contact force is less than or equal to zero.

8 Discussions and future work

As evident from the presented theory and simulation results, motion planning on general uneven terrain is associated with some unique fundamental complexities. Due to non-availability of closed form expressions of the inverse dynamics of a mobile robot on an uneven terrain, it is generally not possible to obtain an one shot stable trajectory connecting the start and the goal state. It is only possible to ascertain the dynamic stability along a candidate path and if stable obtain a stable velocity and acceleration profile for it. Thus, the planner needs to explore various candidate paths before arriving at the final stable trajectory. The proposed work however does provide heuristics which can guide the deformation of the path towards stable regions. As shown in Sect. 2 various non-holonomic candidate paths can be obtained by just solving a set of linear equations, which is computationally simple. The process of ascertaining the stability of a candidate path, as described in Sect. 4 just involves solving a set of linear inequalities or simply searching to obtain a single parameter called the scale factor. This is the pivotal and fundamental contribution of this work. That a complex multi dimensional problem in the space of linear and angular velocities has been reduced to solving a set of linear inequalities for a single parameter has not appeared elsewhere. The process of obtaining a stable time optimal velocity and acceleration for a given scale factor profile requires solving a non-linear optimization problem and on an average takes around 100 ms. Moreover, although non-linear and non-convex, this optimization can be reformulated in a difference of convex form and thus can be solved efficiently. Further the optimization for computing velocities and accelerations needs to be computed only once after the path is ascertained to be stable and a scale factor is obtained at each point along the path. The overall iteration that the planner takes to obtain a stable trajectory depends entirely on the complexity of the underlying terrain. For some difficult terrains multiple trajectory re-planning as shown in example 7.3 might be necessary.

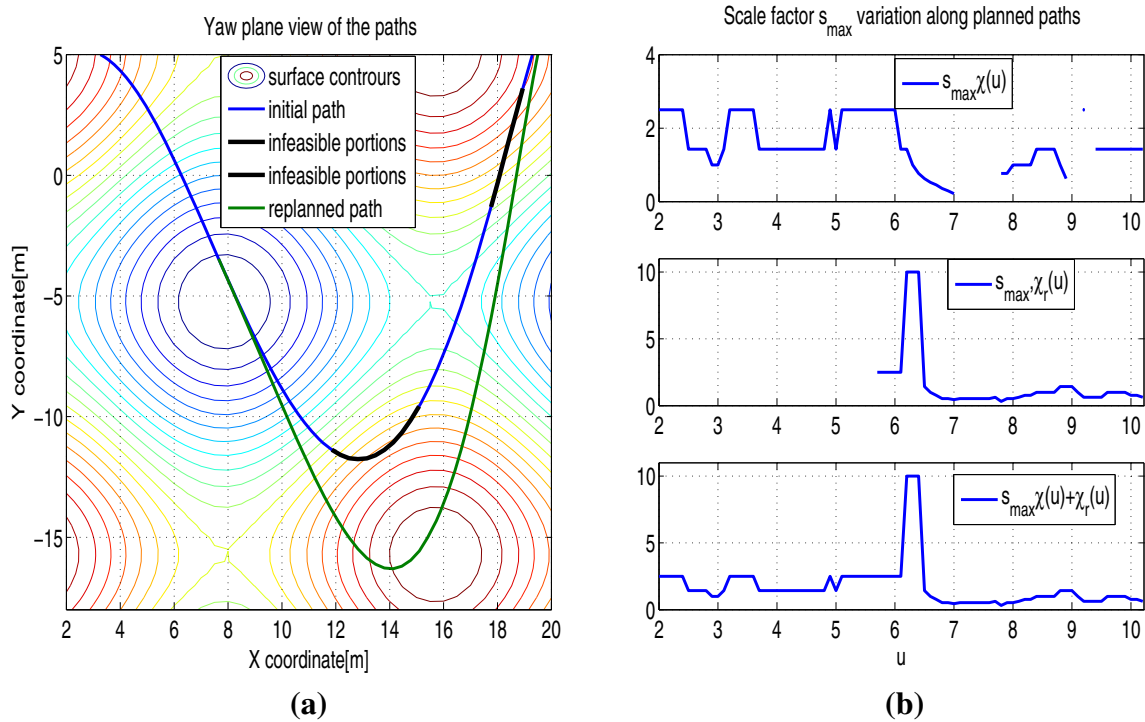


Fig. 15 **a** Figure shows the yaw plane view of the initial path $\chi(u)$. The infeasible portions are shown in black. Infeasible/Unstable portions of the path are re-planned and are shown in green. **b** Scale factor variation along the initial path, re-planned portions and the combined path

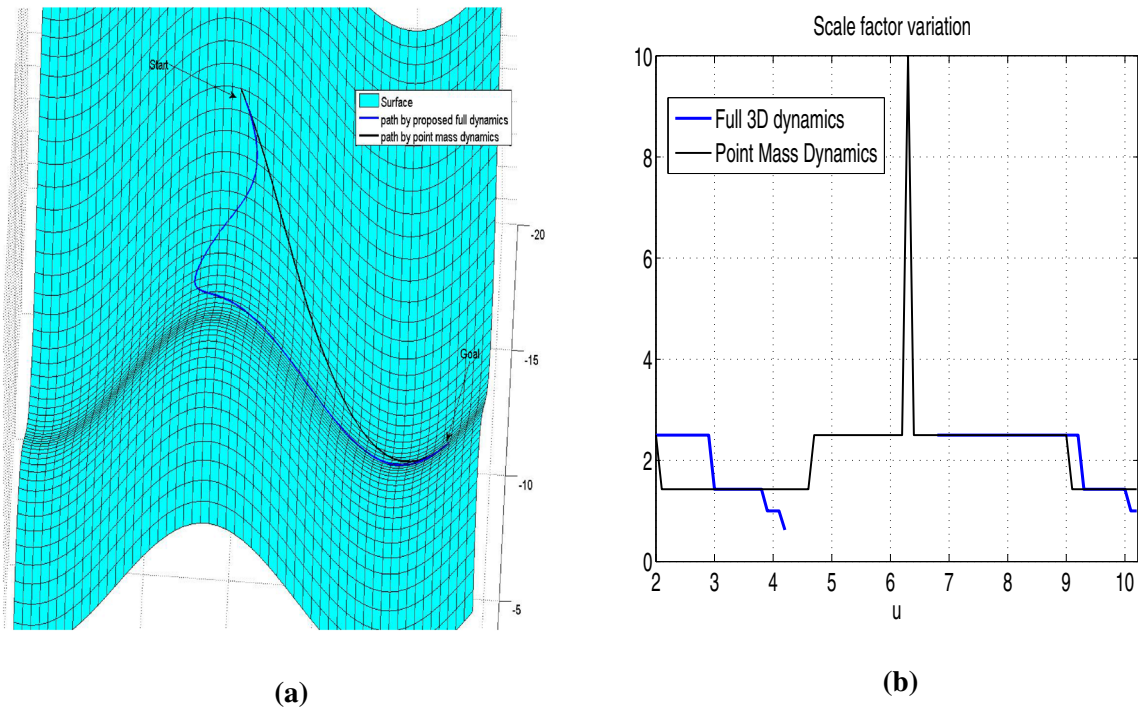


Fig. 16 **a** Figure shows the path comparison based on point mass and full 3D dynamics. **b** Figure shows the scale factor variation when the full vehicle model is evolved along the path computed using the point mass dynamics. It can be seen that if the full 3D dynamics of the vehicle is considered, at various points along the point mass dynamics based

path, scale factor could not be obtained indicating that the no-slip and permanent wheel-ground contact constraints are not satisfied. It can be inferred that point mass dynamics does not capture the true stability picture of the vehicle

As evident, the planner does not include lateral forces while calculating the stability of the robot and the trajectories are obtained with the assumption of no side-slip. This limitation primarily arises out of the fact that side-slip is purely reactive phenomenon and to the best of author's knowledge, there does not exist any mathematical model which relates the applied control input to the side slip that the robot exhibit on a general uneven terrain. This kind of assumption is present with various trajectory generation frameworks like Shiller and Gwo (1991) and Bonnafous et al. (2001). But given the simplicity of the process described in Sect. 4, it is possible to modify the scale factor and hence the velocity and acceleration profile on-line based on the estimated side slip of the vehicle. It is to be noted the magnitude of the lateral force resulting from the side slip will be known and hence it's incorporation does not change the nature of inequalities obtained in the 4.

There are various future directions for the proposed work. The proposed methodology can be extended to four-wheeled active suspension robots. The 6 dof evolution framework proposed in the current work can be directly extended to active suspension robots. The roll, pitch and height of the robot's center are obtained as a function of robot's leg length which will be an additional control input in the case of an active suspension robot. It will be interesting to note the effect of robot's articulation on the nature of the obtained trajectories. The second future line of work dwells with implementing the proposed framework on an experimental platform. To improve the robustness of the uneven terrain navigation, a framework which incorporates the uncertainty in the terrain knowledge is currently being developed. Assuming complete and exact information of the robot's geometry, state and control, the stable velocity and acceleration profile as derived in this paper will be influenced only by the knowledge of the terrain equation. Representing the uncertainty in the terrain equation through a distribution, a probabilistic model is being integrated in the current work, so that a measure of risk along with stability can be obtained for any candidate path.

Appendix 1

$$\eta_i = k_1 l_i \sin \alpha - k_2 l_i \cos \alpha \quad (59)$$

$$v_i = k_1 l_i \cos \alpha + k_2 l_i \sin \alpha \quad (60)$$

$$H_1 = k_1 x - k_1 w \sin \alpha - k_1 h \cos \alpha + k_2 y - k_2 h \sin \alpha + k_2 w \cos \alpha + k_3 + l_1 \quad (61)$$

$$H_2 = k_1 x + w(k_2 \cos \alpha - k_1 \sin \alpha) + h(k_1 \cos \alpha + k_2 \sin \alpha) + k_3 + l_2 + k_2 y \quad (62)$$

$$H_3 = k_1 x + w(k_1 \sin \alpha - k_2 \cos \alpha) + h(k_1 \cos \alpha + k_2 \sin \alpha) + k_3 + l_3 + k_2 y \quad (63)$$

$$H_4 = k_1 x + w(k_1 \sin \alpha - k_2 \sin \alpha) + h(-k_1 \cos \alpha - k_2 \sin \alpha) + k_3 + l_4 + k_2 y \quad (64)$$

Appendix 2

Inverting the coefficient matrix in (18), and using the relations in (59–64) we get

$$\gamma = k_2 \cos \alpha - k_1 \sin \alpha \quad (65)$$

$$\beta = -k_1 \cos \alpha - k_2 \sin \alpha \quad (66)$$

Differentiating (65) twice we get

$$\begin{aligned} \frac{d^2 \gamma}{dt^2} &= \frac{d^2 k_2}{dy^2} \left(\frac{dy}{dt} \right)^2 \cos \alpha + \frac{dk_2}{dy} \frac{d^2 y}{dt^2} \cos \alpha - \frac{dk_2}{dy} \frac{dy}{dt} \sin \alpha \frac{d\alpha}{dt} \\ &+ \frac{dk_2}{dy} \frac{dy}{dt} \sin \alpha \frac{d\alpha}{dt} \\ &+ k_2 \cos \alpha \left(\frac{d\alpha}{dt} \right)^2 - k_2 \sin \alpha \frac{d^2 \alpha}{dt^2} - \frac{d^2 k_1}{dx^2} \left(\frac{dx}{dt} \right)^2 \sin \alpha \\ &- \frac{dk_1}{dx} \frac{d^2 x}{dt^2} \sin \alpha \\ &- \frac{dk_1}{dx} \frac{dx}{dt} \cos \alpha \frac{d\alpha}{dt} - \frac{dk_1}{dx} \cos \alpha \frac{dx}{dt} \frac{d\alpha}{dt} + k_1 \sin \alpha \left(\frac{d\alpha}{dt} \right)^2 \\ &- k_1 \cos \alpha \frac{d^2 \alpha}{dt^2} \end{aligned} \quad (67)$$

By putting $\dot{x}(t) = x'(u)s_u$, $\ddot{x}(t) = x''(u)s_u^2$, $\dot{y}(t) = y'(u)s_u$, $\ddot{y}(t) = y''(u)s_u^2$, $\dot{\alpha}(t) = \alpha'(u)s_u$, $\ddot{\alpha}(t) = \alpha''(u)s_u^2$, as mentioned in Sect. 4 it can be shown that

$$\ddot{\gamma}(t) = \ddot{\gamma}(u)s_u^2 \quad (68)$$

Similarly

$$\ddot{\beta}(t) = \ddot{\beta}(u)s_u^2 \quad (69)$$

Using (68) and (69) it is straightforward to observe that $\vec{\Omega}(t) = \vec{\Omega}'(u)s_u^2$. The derivation of $\ddot{z}(t)$ in terms of $\ddot{z}(u)$ proceeds along similar lines and is left to the reader.

Appendix 3

For deriving the equations of motion of the vehicle, the wheel ground contact normal and traction force unit vector needs to be calculated. Wheel ground contact normal can be calculated based on the wheel ground contact point information derived in Sect. 3 as:

$$\begin{bmatrix} n_{xi} & n_{yi} & n_{zi} \end{bmatrix}^T = \begin{bmatrix} \frac{f_x}{\sqrt{f_x^2 + f_y^2 + f_z^2}} \\ \frac{f_y}{\sqrt{f_x^2 + f_y^2 + f_z^2}} \\ \frac{1}{\sqrt{f_x^2 + f_y^2 + f_z^2}} \end{bmatrix} \quad (70)$$

$f_x = \frac{\partial(a-f(b,c))}{\partial b}$, $b = x_{ci}$, $c = y_{ci}$, $a = z_{ci}$ $f_y = \frac{\partial(a-f(b,c))}{\partial c}$, $b = x_{ci}$, $c = y_{ci}$, $a = z_{ci}$ Once the unit normal vectors are calculated the traction force unit vector can be derived with the help of wheel axis unit vector which in our case has been taken as

$$\hat{\mu}_i = R \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T \quad (71)$$

$$\hat{t}_i = \frac{\hat{\mu}_i \times \hat{n}_i}{|\hat{\mu}_i \times \hat{n}_i|} \quad (72)$$

With the above information the equations of motion can be written as

$$\sum_{i=1}^4 N_i \hat{n}_i + \sum_{i=1}^4 T_i \hat{t}_i = \begin{bmatrix} F_x & F_y & F_z \end{bmatrix}^T \quad (73)$$

$$\sum_{i=1}^4 r_i \times N_i \hat{n}_i + \sum_{i=1}^4 r_i \times T_i \hat{t}_i = \begin{bmatrix} M_x & M_y & M_z \end{bmatrix}^T \quad (74)$$

$$r_i = \vec{P}_{gci} \quad (75)$$

$$F_x = ma_x \quad (76)$$

$$F_y = ma_y \quad (77)$$

$$F_z = ma_z + mg \quad (78)$$

$$M_x = I_{xx} \dot{\Omega}_x + I_{zz} \Omega_y \Omega_z - I_{yy} \Omega_y \Omega_z \quad (79)$$

$$M_y = I_{yy} \dot{\Omega}_y + I_{xx} \Omega_x \Omega_z - I_{zz} \Omega_x \Omega_z \quad (80)$$

$$M_z = I_{zz} \dot{\Omega}_z + I_{yy} \Omega_x \Omega_y - I_{xx} \Omega_x \Omega_y \quad (81)$$

I_{xx} , I_{yy} , I_{zz} are the moment of inertia of the chassis and here a diagonal Inertia matrix has been taken.

Equations (73) and (74) can be written in the matrix form $A * C = D$ as mentioned in Sect. 3

References

- Amenti, N., Bern, M., & Kamvyselis, M. (1998). A new Voronoi-based surface reconstruction algorithm. In *Proceedings of SIGGRAPH* (pp. 415–421).
- Bajaj, C. L., Bernardini, F., & Xu, G. (1995). Automatic reconstruction of surfaces and scalar fields through 3D scans. In *Proceedings of SIGGRAPH* (pp. 109–118).
- Bonnafous, D., Lacroix, S., & Simeon, T. (2001). Motion generation for a rover on rough terrains. In *Proceedings of International Conference on Intelligent Robots and Systems* (pp. 784–789). Maui, USA.
- Cherif, M. (1999). Motion Planning for all terrain vehicles: A physical modeling approach for coping with dynamic and contact interac-

- tion constraints. *IEEE Transactions on Robotics and Automation*, 15(2), 202–218.
- Gallina, P., & Gasperatto, P. (2000). A technique to analytically formulate and to solve the 2 dimensional constrained trajectory planning problem for a mobile robot. *Journal of Intelligent and Robotic Systems*, 27(3), 237–262.
- Guo, Y., Long, Y., & Sheng, W. (2007). Global trajectory generation for nonholonomic robots in dynamic environments. In *Proceedings of International Conference on Robotics and Automation* (pp. 1324–1329). Rome, Italy.
- Hollerbach, J. M. (1984). Dynamic scaling of manipulator trajectories. *ASME Journal of Dynamic Systems, Measurement, and Control*, 106(1), 102–106.
- Howard, T. M., & Kelly, A. (2007). Optimal rough terrain trajectory generation for wheeled mobile robots. *International Journal of Robotics Research*, 26(2), 141–166.
- Iagnemma, K., Shimoda, S., & Shiller, Z. (2008). Near-Optimal Navigation of High Speed Mobile Robots on Uneven Terrain. In *Proceedings of International Conference on Intelligent Robots and Systems* (pp. 4098–4103). Nice, France.
- Inanc, T., Bhattacharya, K., & Muezzinoglu, M. K. (2005). Use of parametric approximation in real-time non linear trajectory generation. In *Proceedings of Conference on Decision and Control* (pp. 6808–6813). Seville, Spain.
- Jones, J., Ihrampetakis, N. P. & Piigl, A. C. (1996). Computation of the generalized inverse of a rational matrix via maple and applications. In *International Symposium on Computer Aided Control System Design* (pp. 495–500). Dearborn, MI.
- Kubota, T., Kuroda, Y., Kunii, Y., & Yoshimitsu, T. (2001). Path planning for a newly developed microver. In *Proceedings of International Conference on Robotics and Automation* (pp. 3710–3715). Seoul, Korea.
- Mann, M., & Shiller, Z. (2008). Dynamic stability of offroad vehicles: A quasi-3D analysis. In *Proceedings of International Conference on Robotics and Automation* (pp. 2301–2306). Pasadena, CA.
- Miro, J. V., Dumonteil, Beck, G. Ch., & Dissanayake, G. (2010). Kino-dynamic metric to plan stable paths on rough terrains. In *Proceedings of International Conference on Intelligent Robot and Systems* (pp. 294–299). Taiwan, Japan.
- Nagy, B., & Kelly, A. (2001). Trajectory generation for car-like robots using cubic curvature polynomials. *Field and Service Robots*, 11.
- Prokos, A., Karras, G., & Petsa, E. (2010). Automatic 3D surface reconstruction by combining stereo vision with the slit-scanner approach. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XXXVIII, Part 5 Commission V Symposium, Newcastle upon Tyne, UK.
- Shiller, Z. (2000). Obstacle traversal for space exploration. In *IEEE International Conference on Robotics and Automation, 2000 (ICRA'00)* (Vol. 2, pp. 989–994). IEEE.
- Shiller, Z., & Gwo, Y.-R. (1991). Dynamic motion planning of autonomous vehicles. *IEEE Transactions on Robotics and Automation*, 7(2), 241–249.
- Singh, A. K., Krishna, K. M., & Eathakota, V. (2011). Planning stable trajectory on uneven terrain based on feasible acceleration count. In *Proceedings of Conference on Decision and Control* (pp. 6373–6379). Miami, FL.
- Singh, A. K., Krishna, K. M., & Saripalli, Srikant (2012). Planning trajectories on uneven terrain using optimization and non-linear time scaling techniques. In *Proceedings of Intelligent Robots and Systems* (pp. 3538–3545). Vilamoura, Portugal.
- Singh, A. K., & Krishna, K. M. (2013). Coordinating mobile manipulator's motion to produce stable trajectories on uneven terrain based on feasible acceleration count. In *Proceedings of Intelligent Robots and Systems* (pp. 5009–5014). Tokyo, Japan.
- Singh, A. K. (2015). Planning with differential constraints: Application to navigation of wheeled mobile robots

on uneven terrain and in dynamic environments. Ph.D. dissertation, International Institute of Information Technology, Hyderabad. http://web2py.iiit.ac.in/research_centres/publications/download/phdthesis.93995963d47d17b3.6172756e636f6d627468657369732e706466.

Waheed, I., & Fotouhi, R. (2009). Trajectory and temporal planning of a wheeled mobile robot on an uneven surface. *Robotica*, 27(4), 481–498.

Wang, H., Kearney, J., & Atkinson, K. (2002). Arc-length parameterized spline curves for real-time simulation. *5th International Conference on Curves and Surfaces*, Oslo, Norway.



Arun Kumar Singh is a PhD candidate in Robotics Research Centre, IIIT-Hyderabad, India. His research interests include motion planning with differential constraints with application to uneven terrain navigation and multi-robot trajectory generation. From 2009-2010 he was associated with Aerospace Engineering Department, IISc - Bangalore, India. He received his Bachelors in Mechanical Engineering from NIT Durgapur, India in 2009.



His research interests include mobile robot motion planning, SLAM, and outdoor navigation of mobile robots.



Srikanth Saripalli is an Assistant Professor at School of Earth and Space Exploration, Arizona State University where he also heads the robotics group ASTRIL. He obtained his Bachelors in Mechanical Engineering from Birla Institute of Technology, Pilani, India in 1999 and his Masters and PhD in Computer Science from University of Southern California in 2002 and 2007 respectively. His research interests include control of unmanned aerial vehicles,

vision based control, visual servoing and control of complex hybrid systems.